Toward Real-time Network-wide Cyber Situational Awareness

Tomas Jirsik*†, Pavel Celeda*
*Institute of Computer Science, †Faculty of Informatics
Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic
E-mail: {jirsik,celeda}@ics.muni.cz

***Abstract*—In today's complex computer networks, we are constantly facing a risk of data loss, system compromise, or intellectual property theft. The complexity of the networks hinders their effective defense. A Network-wide Cyber Situational Awareness (NwCSA) has been introduced to assist a network security administrator with network security. The concept, however, faces several challenges that hinder an efficient application of the NwCSA in a real-world environment. The challenges include the overload of raw data, low speed of reaction, and a lack of context and unified view on a network. In this paper, we present a novel framework that faces above mentioned challenges. The framework leverages a distributed data stream processing system and methods for real-time big data processing. The framework is evaluated with respect to stated requirements on systems for NwCSA. Moreover, we present a prototype framework implementation and provide lessons learned from its real-world deployment.**

I. Introduction

Computer networks continue to increase in their sophistication and complexity. Various commercial services, critical systems, and information sources are maintained in such networks, which makes them a lucrative target for terrorists, cyber espionage, or criminal activities. The intruders take advantage of the complexity and sophistication of the networks to elude the security measures. To be able to defend systems from the intruders, it requires an ability to retrieve data about the network, comprehend processes in the network, identify and detect a threat, and take appropriate actions, i.e. to achieve a situational awareness over a network.

Situational awareness (SA) as a general concept is widely discussed in literature [1], [2]. The situational awareness is defined in [1] as "perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future". The application of the SA concept in the network security domain is, however, not straightforward. The efficient application of Network-wide Cyber Situational Awareness (NwCSA) faces several major challenges due to high dynamics of the network environment, speed and volume of network traffic, and complexity of the computer networks. The first challenge is an overabundance of raw data. An analyst is overloaded with the raw network data [3]. Volume, velocity, and variability of the raw data prevent him/she them from comprehending a network and taking proper decisions. The next challenge is the speed at which cyber events occur. The reaction speed of defenders is much smaller in comparison to the speed of the intruder's actions due to automation of the attacks. Last but not least challenge is to provide a homogeneous toolset with a unified view both on whole network and individual elements in a network.

In this paper, we first provide necessary background by defining NwCSA and describing relevant state-of-the-art. We focus on a detailed elaboration of possibilities of network perception and comprehension. Based on the elaboration, we identify requirements for an efficient application of NwCSA that respond to above-mentioned challenges. Next, we present a novel framework that meets the identified requirements. The framework reduces data overload through distributed data computing suitable for processing of a large volume data. The speed of the reaction is improved by a shift from a traditional batch-based data analysis to a stream-based approach. Further, we present an innovative solution for unified presentation of information about both whole network and individual network elements. The efficiency of the framework is demonstrated on a real-world prototype deployment. The lessons learned from the deployment are offered in the discussion section.

II. Network-wide Cyber Situational Awareness

The SA has been originally used in conventional military conflicts to assist officers in taking strategic decisions. With a transition from conventional battlefields to cyber ones, the traditional SA had to be adapted to the cyber environment. The traditional SA has evolved into Cyber Situational Awareness (CSA). Network-wide Cyber Situational Awareness (NwCSA) is a specific field of CSA which focuses on gaining SA in computer networks. The elements considered in NwCSA are parts of network physical infrastructure (i.e. switches, routers, hosts, servers, and lines), and network traffic transferred via a network. The goal of NwCSA is to provide the insight into the dynamics of computer networks and provide information necessary to answer questions arising in the decision process, e.g. where did an attacker come from, and what harms have been caused by an attack.

NwCSA systems are multilevel. A NwCSA system relies on information from intrusion detection systems (IDS), antiviruses, malware detectors, logs, flows and other information sources. This raw information is transformed into events that are further processed [4], [5]. The number of events is however still too high, and their processing is too labor-intensive to be processed manually. The systems for automatic creation of even higher abstractions are needed [3]. The higher abstractions are, for example, graphs of networks with vulnerability dependencies, or decision trees serving as cyber defense support. Prototype systems that are capable of providing even higher abstractions are emerging in literature, e.g. CAULDRON [4], AHEAD [6]. Further, the distillation of valuable information for NwCSA from network data becomes a big data problem as the volume of network traffic is increasing on a compound annual growth rate of 21 % every five years [7].

The challenges of NwCSA covered by this paper, i.e. overabundance of raw data, unified view, and speed of reaction, are relevant to perception and comprehension part of the NwCSA. In following subsections, we present state-of-the-art of the collection methods and tools that serve for network perception along with approaches to gaining network comprehension.

A. Network Perception

The network perception represents a collection of information about network status, attributes, and dynamics. Raw information from a network is collected via *probes*. There exist two types of probes - active and passive. The passive probes are transparent to the network and collect data passing through an observation point. Active probes, on the other hand, actively gather information from a network. They probe a network and based on the response they determine the information output, e.g. what hosts are present in a network.

There are two types of raw data, which are observed. We observe data about the network itself (e.g. type of devices, their position in a network, links, routing) or we gather information from network traffic (e.g. who is communicating with whom, how long). For observation of network itself, mainly active probes are used (Nmap [8], ZMap [9] or commercial tools Network Topology Mapper [10], IPsonar [11], and WhatsUp-Gold [12]). Further, information about the network infrastructure can also be collected using simple network management protocol (SNMP) [13] or via a centralized collection of logs from network devices, e.g. rsyslog [14].

Passive probes are used mainly for observation network traffic. The network traffic offers insight into the actual behavior of network elements. It shows, e.g., who is communicating with whom, when, and for how long. Available raw data sources for network perception are network packets, IP flows, and logs. The packet analysis offers the most detailed information we can obtain from network traffic. However, the volume of the packets and data to analyze is enormous, and resources needed for analysis are excessive. Therefore, the packet analysis is not efficient on a network-wide scale, and it is mostly used on demand in specific cases. The tools for capturing packets from networks are e.g., tcpdump [15], tshark [16].

IP flows are used mainly for a network-wide traffic monitoring. An IP flow is an abstraction of a network connection defined as a set of packets with common properties passing an observation point during a certain interval. [17]. Due to the abstraction, a volume of data to analyze is lower than in the case of packet analysis. For example, in a medium-sized network of 24,000 active IP addresses, we observed an average of 12,000 flows/second and 110,000 flows/second in the national wide research and education network. Open-source IP flow probes are e.g. fprobe [18], YAF [19], and nprobe [20].

B. Network Comprehension

Comprehension phase of NwCSA process covers an understanding of information carried in a raw data. From the raw data, we derive advanced relations such as critical points or bottlenecks in a network infrastructure, top talkers, suspicious behavior, vulnerabilities, and so forth.

After the collection during network perception, the raw data are stored then into collectors. Type of incoming data determines the type of a collector and way of data storage. IP flow collectors (e.g. nfcapd [21] or IPFIXcol [22]) store data in binary format into column databases into a usually five minutes bins. Dumps from network traffic are stored in a libpcap file format into .pcap files and logs are typically stored in Hadoop Distributed File System (HDSF) [23]. The outputs from active probes are stored in various arbitrary formats.

The analysis of raw data a set of queries and responses on data stored on a collector. Raw network traffic dumps are analyzed by an individual .pcap file. Analysis of the raw traffic is done ad-hoc and generally on demand. IP flows stored on collector are typically analyzed in half- to five-minute batches. The analyses are run either on a regular basis, e.g. every five minutes or on demand, e.g. analysis of a host's communication in a network. The analysis of network logs is done on the fly due to the recent development of data stream processing architectures. The results of the active probes take a form of unstructured text. Automatic processing is then tool-custom. The results are analyzed periodically to update the status of a network continuously.

The analysis workflow influences the speed of reaction in NwCSA. NwCSA aims to lower the reaction time as much as possible [3]. One way how to reach a lower the response time in NwCSA is to reduce a time needed for analysis. As shown in the previous paragraph, network data analyses exhibit different time granularity. The different and rather long analysis times reduce the speed of reaction in NwCSA. Analysis results need to be synchronized among data sources and the fast analyses wait for the results of the slow ones. Even a delay in orders of minutes may cause serious harms to network infrastructure and services worth thousands of dollars [24].

There already exist solutions that are designed aim administrator with comprehend network processes. Commercial solutions are represented by Splunk [25] and AleinVault [26]. However, these solution faces the same challenges as NwCSA does. According to user reviews, AlienVault suffers from performance issues when handling a large volume of logs [27], and Splunk can be enhanced in support of additional types of databases [28].

C. Requirements

A great variety of information is needed to achieve an efficient NwCSA. Figure 1 summarizes the current state of network perception and comprehension described in previous subsections and highlights issues of contemporary NwCSA.

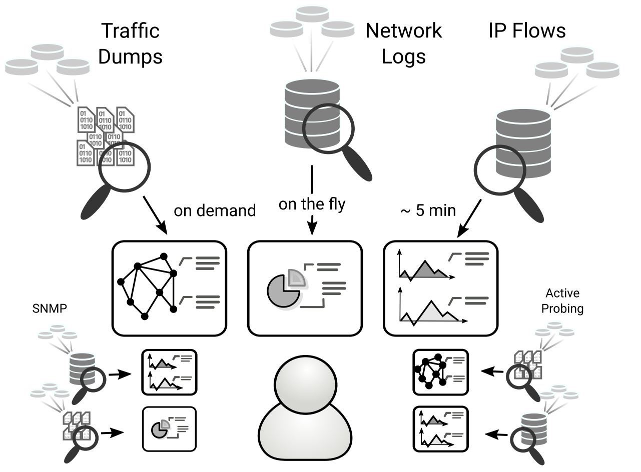

Fig. 1. Current state of perception and comprehension of a network.

First, there exist many tools that retrieve data of various types from a network. Each tool usually uses own storage and leverages a tool-specific analysis language and approaches. The tools have different settings of data collection process and storage which influences analysis workflow and data comprehension. Second, raw data is collected at high speeds and volumes. As the tools for data collections differ, the data type, format, frequency, and information they carry differ too. Moreover, information carried in raw data can overlap, duplicate, or even contradict, which further impedes the analysis. Third, a network administrator has to understand many network monitoring methods and approaches, switch between them for specific information, run different types of analyses, and use various levels of details in an analysis. The diversity of data sources and complexity of the analysis also hinders the decision process as it takes to an administrator a longer time to collect all data necessary for an informed decision. The result is then a low reaction speed in NwCSA.

To face above mentioned disadvantages and challenges in perception and comprehension of a network, we impose requirements for an effective framework for NwCSA:

- **Performance** - the framework should be able to process and analyze large volumes of the data at high speeds.
- **Universality** - the framework should be able to gather and process several data from various data sources.
- **Context** - the framework should be able to offer complete information including context relevant to the information instead overwhelming a user with a flood of raw data.
- **Dynamic level of detail** - the framework should be able to provide a dynamic level of detail both in time and information domain.
- **Reaction time** - the framework should minimize the time needed for analysis to increase the speed of reaction.

In the following section, we apply all requirements and describe a novel framework for a network perception and comprehension in NwCSA along with its prototype implementation.

III. Real-Time Network-wide Cyber Situational Awareness Framework

The proposed framework for network perception and comprehension leverages new advances achieved in distributed data stream computing and apply their concepts to the Network-wide Cyber Situational Awareness domain. Such an approach results in several improvements in performance, universality, and data analysis.

A. Architecture

The proposed framework is depicted in Figure 2. The data are captured from a network via probes. Captured data are not sent to specialized collectors by a data type as in the previous approach, though. Instead, all data from probes are processed in one system. To be able to process data from various probes in one system, the data needs to be normalized into a general format. Normalization takes place in a normalization system (e.g. Logstash [29]). After the normalization system receives data from probes, it decodes them and transforms them into a general Data Serialization Format (DSF), so that the data can be processed in one system. Normalized data is further sent to a messaging system. The purpose of a messaging system is to distribute data for distributed stream processing framework efficiently. There are many messaging systems such as ActiveMQ, RabbitMQ, or Apache Kafka (for the full list see [30]). Currently, the most suitable system is Apache Kafka as it offers sufficient message throughput and is compatible with most data stream processing frameworks.

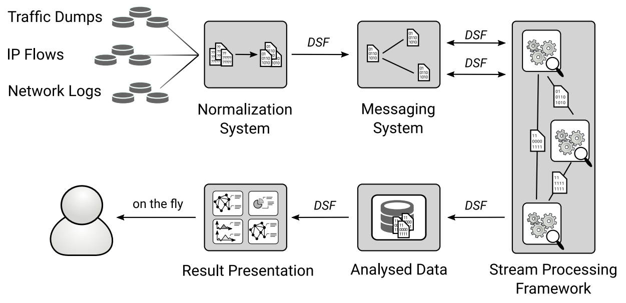

Fig. 2. Framework for perception and comprehension of a network.

The core of the framework is a distributed stream processing system. Systems for distributed data stream processing are capable of processing a large volume of data at high speeds, as has been shown in [31]. The distributed systems can easily adapt to an increase in data volume, speed, and variety.

An additional computational node can be added instantly to the system to boost the performance. Previously impossible analysis, such as computing host statistics for each host from IP flows simultaneously, are now possible. Broadly used systems for distributed data stream processing are Spark, Storm, Samza, and Flink, all maintained by Apache Foundation. These systems provide all basic functional requirements for distributed stream data processing such as data reliability, fault tolerance, and generality. The stream processing systems process data in data streams which enables parallelization of the analysis and increases the speed of analysis. The data is processed on-the-fly immediately after it is received by the systems. The on-the-fly processing makes the real-time data analysis possible and significantly improves a response time of the framework.

Analysis of data in data streams is based on continuous queries. An analysis is initialized by raising the query. Since then, all incoming data are analyzed using the query. An analysis is done in primary memory to achieve a high throughput (no I/O operations are needed). Only results of the analyses are handed in for further processing and storage which improves analysis response time. The processed and analyzed data is then sent to a data storage. A data storage keeps results of data analysis and serves as a data source for result presentation to the user. So-called ”Next Generation Databases” are suitable for NwCSA work-flow. These storages are able to hold a large volume of the data and support advanced queries over the stored data needed for a proper result presentation and comprehension. Currently, the most common database for storing a large volume of data from data stream processing systems is Elastic Stack [32] comprising of Logstash, Elasticsearch, and Kibana.

Result presentation layer is necessary for data comprehension in NwCSA. In the proposed framework, the results of an analysis are presented to a user in a unified manner. Due to the unified processing of the data, the framework can provide information from various data sources in one user interface.

B. Prototype Implementation

To demonstrate a pilot implementation of the framework, we introduce a publicly available prototype *Stream4Flow*[1] [33]. To highlight the advantages of the framework, we implement NwCSA framework based on IP flow data source. An instance of Stream4Flow currently provides a decision support information about a network of 25 000 hosts to a computer incident response team.

The normalization component of the framework, IPFIXCol [22], transforms IP flows to a JSON format. JSON formatted IP flows are distributed into data stream processing system using Apache Kafka. Apache Spark Streaming was selected as the data stream processing system as it provides sufficient data throughput, a wide range of programming languages (Scala, Java, Python), and support for Map-Reduce programming model. Moreover, it offers a GraphX library for graph analysis. Analyzed data are stored in Elasticsearch. Another part of Elastic Stack, Kibana, is used for result preview. The *Stream4Flow* framework also provides an additional web interface (see Fig. 3) that offers both macro and micro view of the network using a combination of a zoomable, clickable heatmap (macro view) and basic visualizations of IP flows characteristics (micro view).

[1] https://stream4flow.ics.muni.cz/

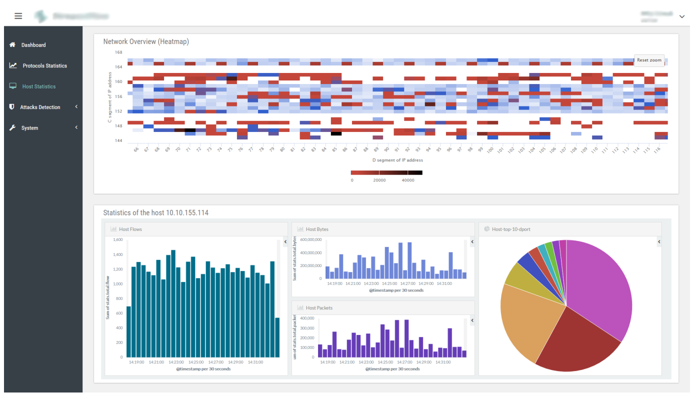

Fig. 3. *Stream4Flow* web interface with macro and micro view

The prototype has been receiving IP flows at average speeds of 8500 IP flows per second for more than two months in the production deployment. Even though the prototype collects only IP flows, it provides a variety of information needed to achieve NwCSA. Basic characteristics of the whole network are computed, as well as individual characteristics for each of 25 000 active hosts in the network, such as the sum of packets, bytes, flows, port utilization, the number of communication peers and so forth. These characteristics have been computed every 10 seconds for each device. The members of the cyber security incident response team have reported a positive effect on understanding the network.

IV. Discussion

In this section, we discuss each of the stated requirements for a NwCSA framework. We examine the framework from performance, universality, dynamic level of detail, context, and reaction time point of view. Conclusions of the discussion are based both state-of-the-art findings and own experiences from prototype deployment. The section concludes with further remarks that describe additional comments and lessons learned applicable for framework deployment.

A. Performance

An efficient framework should be able to process data from variable data sources at high speeds and volumes. Distributed stream processing system used in framework allows data processing even in volumes and velocity described above. In a case of need for higher performance, an additional computational node can be added to increase throughput. Moreover, there is no single point of data input that might become a potential bottleneck. The data are received through multiple input points, which further increases the throughput.

It has been shown in [31] that tools for distributed stream data processing are capable of processing data in volumes and speed over 1M records/second in a small cluster of four commodity servers. The performance requirements on the data storage are further significantly reduced due to the fact, that only preprocessed results are stored instead of raw data. In case a raw data are required for analysis, another distributed storage node for raw data can be added to the framework.

The proposed framework also brings performance advantages regarding data analysis. Some advanced analyses, such as data clustering, computation of long-term characteristics, or top N characteristics, are computationally intensive and demand large volumes of memory and computational power. The distributed nature of stream processing systems allows task parallelization using MapReduce programming principle [34] which makes the computation of such tasks possible. Processing data in data streams also brings a performance advantage. The operations over data streams, such as duplication, split, union, enables us to parallelize the data streams and their analysis. The parallelization further increases the speed and possibilities of the analyses.

B. Universality

Universality requirement guarantees that a framework based system can receive and analyze data from various types of data sources. The universality is achieved in two steps. The first step is the ability to receive from different data sources. The second step is the ability to analyze various data within one system.

Data sources provide data with different structure and information. A normalization systems used in framework maintains universality by using so-called codecs. A codec is a description of a data structure sent by a data source. It specifies the position and meaning of a given information in sent data. Normalization tools, such as Logstash, support input codecs for a variety of data sources including a Netflow v5/9 codec for data from IP flow monitoring, nmap codecs for results active network scanning, or syslog codecs for machine logs.

Various types of data can are analyzed in the proposed framework due to a unified internal representation of the data. An implemented normalization process transforms the data into readable, comprehensible format. Due to a dynamic and structure of received data, a general data serialization format (DSF) is used. Widely popular DSF is Javascript Object Notation (JSON) format. JSON is an open-standard file format that transforms objects into key-value pairs of human-readable text. The DSFs enable to share the data efficiently among the individual components of the framework no matter of the input data format.

C. Dynamic Level of Detail

A comprehension of a network requires two kinds of dynamic details - a dynamic level of time granularity and dynamic level of view perspective. The proposed framework implements both of the dynamic levels of details.

A challenge of dynamic level of time granularity is to provide short-term results. The long-term statistics can be derived from short-term ones by aggregation. The short-term results require a data collection and analysis in small time intervals (so-called micro batches). The stream processing approach used in the framework is designed to process and analyze data in micro batches. The micro batches enable us to achieve a required level of minimal time granularity. The stream processing frameworks also implement a concept of sliding windows suitable for long-term analysis. An illustrative example of the advantages of stream processing framework is an analysis of IP flows. A probe exports IP flows for analysis in a continuous stream. Current tools for IP flow analysis, however, aggregates this continuous stream into batches of five-minute data. The analysis is then done per batch. Using the stream processing frameworks, we can analyze the data per IP flow.

Dynamic level of view represents the possibility to see a network both from overview and in detail. In current NwCSA, both views were obtained by a different set of tools specialized for a given view. Providing both views from raw data in a unified tool was too computationally demanding as raw data for both views needed to be processed. Distributed systems and MapReduce programming principle enable such computations, which makes a macro and micro view in a single tool possible. We can assign an entity identifier (e.g. IP address, MAC address) as a mapping key. Computations of required characteristics, statistics and analyses are then distributed among the machines according to the key. Using this approach, we are able to monitor in detail all entities in a network and, at the same time, monitor the overall network state.

D. Context

Thanks to the successful implementation of performance and universality requirement, the framework meets also the context requirement. The universal nature of the framework enables us to process data from various sources. Scalability and distributed nature of the framework guarantees a sufficient computational power to process and analyze the data. Thus, all necessary raw data needed for NwCSA can be managed in one system.

The possibility of processing all types of data in one framework enables us to combine pieces of information from different data sources effectively. Information about a host network behavior gained from IP flow analysis can be supported by relevant log records from the host. The results of analyses of different data sources can be correlated which each other to increase the precision and robustness of the analyses.

Data stream processing systems are suitable for data preprocessing as they implement continuous queries. Using the continuous queries, we can precompute several predefined characteristics and statistics. The administrator does not need to run analyses over collected raw data. Instead, the analyses are run on preprocessed characteristics. This approach reduces the data overload in NwCSA as an administrator handles preprocessed data instead of raw data.

E. Reaction time

Proposed framework improves the reaction time in two ways: it reduces analysis response time, and it enables a real-time data processing. Both enhancements shorten a time needed for a delivery of information necessary for a decision to an administrator, which decreases the reaction time.

The analysis response time is reduced due to the use of distributed systems for data processing. As described earlier, distributed systems enable parallelization of analysis computation, e.g. by using MapReduce programming model. An analysis can then finish in a shorter time and results of the analysis are available earlier. The analysis response time is also improved by a data preprocessing done by the stream processing system. The analysis process does not need to process a large volume of raw data. Instead, some partial results are precomputed, and only a reduced volume of data is analyzed.

Data stream processing systems included in the framework are able to process data in real time. A piece of a raw information is processed immediately when a stream processing system receives it. Real-time data processing is a significant advantage over current tools for network perception. As described earlier, there are delays caused by the analysis of the data in batches. In the case of IP flows, the delay can reach up to 10 minutes. Including systems capable of real-time data processing into NwCSA framework eliminates such a delays and improves the reaction time.

F. Further Remarks

Setting the real-time distributed stream processing system as the cornerstone of data analysis in NwCSA has following consequences. The nature of the real-time stream processing transforms the approaches to data analysis. In current tools for network data analysis, data are analyzed ex-post. Raw data are stored and then analyzed retrospectively. It is possible to perform a query over historical data or search back the data for additional information if needed. In the stream-based approach, the data cannot be analyzed ex-post. The data are analyzed in data streams using continuous queries. Only results of the queries are stored. Nevertheless, there might occasionally be demand for ex-post analysis. In that case, we recommend extending the framework with a suitable primary data retention store that makes on-demand ex-post analysis possible.

Support of various data sources and processing different data in one system opens a new issue regarding information duplication. In a real-world deployment, the data collection area of the probes can overlap. Two separate probes can then observe the same information. A suitable example is an IP flow that is routed via two probes or the log records from two different machines that represent the same network scan. Duplication of collected information needs to be kept in mind during data analysis and comprehension. In case information is not deduplicated, biased or incorrect findings may occur. The bias is then carried further in NwCSA framework, and misleading decisions are taken.

V. Summary

In this paper, we introduce a novel framework for network-wide cyber situational awareness that aims to face current challenges of NwCSA. The framework takes advantages of recent advances in distributed data stream processing. The considerable computational power of these systems enables us to analyze all data harvested from a network in one system, which was not possible before. The stream-based data processing also reduces the time needed for reaction as data are processed on the fly. The framework reduces a volume of data that an analyst needs to analyze as only results of the preprocessing are stored for analysis by the administrator. The universality of the framework is ensured by a normalization component that transforms raw data into a common representation format. Efficient result analysis and data comprehension are possible due to the usage of next-generation databases and novel visualizations.

Presented framework surpass current solutions for NwCSA by providing a real-time overview of a network in one system, supplying administrators with both general and detailed information about a network, and by adding context to network data. The framework is discussed regarding performance, universality, dynamic level of detail, data context, and reaction time. Moreover, experiences from a real-world deployment of the framework are provided.

The framework focuses on network perception and comprehension part of NwCSA. There are still open issues in a projection of network status in the future and cognitive processes in NwCSA. The issues include, but are not limited to the ability to see the likely outcome of a given decision, prediction of attacker's actions, or optimization and capture of cognitive reasoning process of an administrator. Nevertheless, we believe, that the presented framework for NwCSA can be further improved to address the above-introduced issues.

Acknowledgment

This research was supported by the Security Research Programme of the Czech Republic 2015-2020 (BV III/1–VS) granted by the Ministry of the Interior of the Czech Republic under No. VI20162019014 – Simulation, detection, and mitigation of cyber threats endangering critical infrastructure.

References

[1] M. Endsley and E. O. Kiris, "The out-of-the-loop performance problem and level of control in automation," *Human Factors*, vol. 37, no. 2, pp. 381–394, 1995. [Online]. Available: http://dx.doi.org/10.1518/001872095779064555

[2] M. Endsley and D. Garland, *Situation Awareness Analysis and Measurement*, ser. Situation Awareness: Analysis and Measurement. Taylor & Francis, 2000.

[3] A. Kott, C. Wang, and R. F. Erbacher, *Cyber Defense and Situational Awareness*. Springer, 2014.

[4] S. Jajodia, S. Noel, P. Kalapa, M. Albanese, and J. Williams, "Cauldron mission-centric cyber situational awareness with defense in depth," in *2011 - MILCOM 2011 Military Communications Conference*, Nov 2011, pp. 1339–1344.

[5] M. Drasar, T. Jirsik, and M. Vizvary, *Enhancing Network Intrusion Detection by Correlation of Modularly Hashed Sketches*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 160–172.

[6] F. De Gaspari, S. Jajodia, L. V. Mancini, and A. Panico, "Ahead: A new architecture for active defense," in *Proceedings of the 2016 ACM Workshop on Automated Decision Making for Active Cyber Defense*, ser. SafeConfig '16. New York, NY, USA: ACM, 2016, pp. 11–16.

[7] C. and/or its affiliates, "Cisco VNI Forecast and Methodology, 2015-2020," June 2016, [cited 2017-04-27]. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[8] G. F. Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. USA: Insecure, 2009.

[9] Z. Durumeric, E. Wustrow, and J. A. Halderman, "Zmap: Fast internet-wide scanning and its security applications," in *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*. Washington, D.C.: USENIX, 2013, pp. 605–620. [Online]. Available: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/durumeric

[10] solarwinds, "Network topology mapper," May 2017, cited on 02-05-2017. [Online]. Available: http://www.solarwinds.com/network-topology-mapper

[11] Lumeta, "Ipsonar," May 2017, cited on 02-05-2017. [Online]. Available: http://www.lumeta.com/products/ipsonar/

[12] ipswitch, "Whatsup gold network monitoring," May 2017, cited on 02-05-2017. [Online]. Available: https://www.ipswitch.com/application-and-network-monitoring/whatsup-gold

[13] J. Dilley and I. Cooper, "Known HTTP Proxy/Caching Problems," RFC 3143, Jun. 2001. [Online]. Available: https://rfc-editor.org/rfc/rfc3143.txt

[14] R. Gerhards, "Rsyslog," May 2017, cited on 04-05-2017. [Online]. Available: http://www.rsyslog.com/

[15] tcpdump, "tcpdump," May 2017, cited on 02-05-2017. [Online]. Available: https://www.wireshark.org/docs/man-pages/tshark.html

[16] Wireshark, "tshark," May 2017, cited on 02-05-2017. [Online]. Available: https://www.wireshark.org/docs/man-pages/tshark.html

[17] B. Claise and B. Trammell, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011, Sep. 2013. [Online]. Available: https://rfc-editor.org/rfc/rfc7011.txt

[18] S. Astashonok, "fprobe," May 2017, cited on 04-05-2017. [Online]. Available: https://sourceforge.net/projects/fprobe/

[19] C. M. U. CERT, "Yaf - yet another flowmeter," May 2017, cited on 04-05-2017. [Online]. Available: https://tools.netsa.cert.org/yaf/

[20] ntop, "nProbe," May 2017, cited on 04-05-2017. [Online]. Available: http://www.ntop.org/products/netflow/nprobe/

[21] P. Haag, "nfdump," May 2017, cited on 05-05-2017. [Online]. Available: http://nfdump.sourceforge.net/

[22] P. Velan, "Ipfixcol," May 2017, cited on 05-05-2017. [Online]. Available: https://github.com/CESNET/ipfixcol

[23] T. A. S. Foundation, "Hadoop distributed file system," May 2017, cited on 05-05-2017. [Online]. Available: https://hadoop.apache.org/

[24] U. Franke, P. Johnson, J. König, and L. Marcks von Würtemberg, "Availability of enterprise it systems: an expert-based bayesian framework," *Software Quality Journal*, vol. 20, no. 2, pp. 369–394, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11219-011-9141-z

[25] I. Splunk, "Splunk enterprise," May 2017, cited on 05-05-2017. [Online]. Available: https://www.splunk.com/

[26] I. AlienVault, "Unified security management," May 2017, cited on 05-05-2017. [Online]. Available: https://www.alienvault.com/

[27] trustradius, "Alienvault usm reviews," May 2017, cited on 05-05-2017. [Online]. Available: https://www.trustradius.com/products/alienvault/reviews

[28] L. yuan Lai, "Splunk and spark," May 2017, cited on 05-05-2017. [Online]. Available: https://conf.splunk.com/session/2015/conf2015_LYuan_Splunk_BigData_DistributedProcessingwithSpark.pdf

[29] Elasticsearch, "Logstash," May 2017, cited on 016-05-2017. [Online]. Available: https://www.elastic.co/products/logstash

[30] L. Strzalkowski, "Queues," 2016, cited on 12-05-2017. [Online]. Available: http://queues.io

[31] M. Cermak, D. Tovarnak, M. Lastovicka, and P. Celeda, "A performance benchmark for netflow data analysis on distributed stream processing systems," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016, pp. 919–924.

[32] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*, 1st ed. O'Reilly Media, Inc., 2015.

[33] T. Jirsik, M. Cermak, D. Tovarnak, and P. Celeda, "Toward Stream-Based IP Flow Analysis," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 70–76, 2017.

[34] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, ser. OSDI'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 10–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251254.1251264