IEEE Access
practical innovations : open solutions

Received February 5, 2014, accepted February 18, 2014, date of publication February 25, 2014, date of current version March 5, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2308320

Automatic Model Creation to Support Network Monitoring

MARKO MÄÄTTÄ AND TOMI RÄTY
VTT Technical Research Centre of Finland, Oulu 90571, Finland
Corresponding author: M. Määttä (marko.maatta@vtt.fi)

ABSTRACT The large variety of network traffic sets many challenges in modeling the essential aspects of network traffic flows. Analyzing and collecting features for the model creation process from the network traffic traces is a time-consuming and error-prone task. Automating these procedures are a challenge. The research problem discussed in this paper concentrates on the analysis and collection of features from the network traffic traces for the model development process, by automating the analysis and collection. The proposed system of this paper, called MGtoolV2, supports the model development process through the automation of collection and analysis in the actual model creation procedures. The model development process aims to enhance the development of a model by reducing the development cost and time. The proposed tool automatically creates large sets of models according to the network traffic traces and minimizes the errors of manual modeling. The experiments conducted with MGtoolV2 indicate that the tool is able to create the models from the traffic traces cost effectively. MGtoolV2 is able to unify similarities between packets, to create very detailed models describing specific information, and to raise the abstraction level of the created models. The research is based on the constructive method of the related publications and technologies, and the results are established from the testing, validation, and analysis of the implemented MGtoolV2.

INDEX TERMS Network monitoring, modeling, MSC, Pcap, XML.

I. INTRODUCTION

Today's IP (Internet Protocol) networks transmit a vast amount of information, such as transferring files, e-mail, and other application-specific data. To be able to understand the information flow sequences in the network, it is important to have a formal modeling notation that provides an overview of the network traffic. Furthermore, an efficient way to create those models is a necessity. Analyzing the network traffic traces and manually creating the models is laborious and ineffective. Automatic analysis of the traffic traces and a model creation process are required to address this problem.

This paper presents a tool called MGtoolV2 (Model Generator tool Version 2), which can be used to create models from network traffic traces. These traces are often provided in packet capture (Pcap) format, which is considered as the de-facto standard used in network traffic recordings. The novelty of this paper is that it provides a solution for the following research problem: automating the feature collection and analysis of network traffic traces for the model development process. The proposed tool is able to read Pcap traffic traces and creates the traffic models automatically. These automatically created models are used in analyzing the monitored traffic patterns.

The experiments indicate that automating the model creation process decreases the manual effort required. A large and elaborate set of models can be created within a short time period, thus easing the task of creating the models. Another very important aspect is that the manual insertion of errors and misunderstandings is minimized when using the proposed tool. Our work provides a process and a tool to create network traffic models automatically. These models can be utilized in various network-monitoring contexts, and the costs of the required models for network monitoring are reduced.

The rest of this paper is organized as follows. Section 2 describes the background for the paper. Section 3 gives a detailed description of the proposed tool and the model development process. The experimental case for testing the proposed tool is given in Section 4. Section 5 evaluates the proposed tool according to the experiments. The last section concludes the paper with future development plans.

II. BACKGROUND

The MGtoolV2 tool creates the models using Extensible Mark-up Language[1] (XML) and Message Sequence

[1] http://www.w3.org/standards/xml/core

Chart (MSC) [1] notations, and it is further developed from the tool proposed in [2]. A tool called MGtool was originally developed to create a set of XML and MSC models for network intrusion detection using the Snort rules. The Pcap traces were also utilized in creating the models for malicious network activities, such as port scans and denial-of-service attacks.

The main characteristics of the network traffic are needed when designing, controlling, simulating, and monitoring the communication networks [3]–[5]. Describing essential packet flows requires appropriate traffic models [3]–[5]. In our work these network traffic models are constructed using the XML and MSC notations. In this work the term XML model [6] or the term MSC model [7] stands for the network traffic model.

The MGtoolV2 tool has been substantially developed from the previous work [2], where the fundamental differences are the following. The tool is able to handle the relevant traffic features, to transfer those features into XML and MSC models, and to raise the abstraction level of the created model when appropriate. When the abstraction level is increased, the irrelevant traffic features are omitted from the created models. MGtoolv2 creates the XML and MSC models automatically, so that human effort is minimized and a large set of new models is created cost effectively.

The modeling of network traffic has some challenges. The application space ranges greatly and network traffic contains activities from hundreds or thousands of applications that rely on TCP (Transmission Control Protocol) [8]. The ability to collect the characteristics of the network traffic enables the modeling of the network traffic, and this is considered the most basic function in network planning and operation monitoring [9]. This, combined with the collection of the network traffic from different links, varies enormously [8] and sets another challenge for automatic model creation. However, the network traffic captured from a single link is usable [10]. It can be used to create models according to the network activities monitored from the respective link [10]. The combination of data from multiple links enhances network model creation [11].

The features that are collected from the monitored network traffic for model creation depends greatly on the application domain in which the models will be deployed. Extracting the relevant features for model creation is becoming more urgent [12]. The most common features that are collected from network traffic are the source IP address, destination IP address, source port number, destination port number, and used protocol (such as TCP, UDP (User Datagram Protocol), or ICMP (Internet Control Message Protocol)) [11], [13], [14]. These and packet size information are important for collecting the network traffic flow information [15]. Extracting only the relevant features lowers the costs and the resource consumption [16]. The challenges are how to identify and collect the fundamental features, how these fundamental features are transformed into the model [16], and how to preserve the modeling paradigm [17].

In general, XML is supported by several applications, and developing new applications is simple because XML is concise and formal. These include the definition of logical rules for decision-making [18], simulation modeling [19], and representation of hierarchical models for application development [17]. The XML notation can be adopted in network environments. It has been used for describing the network specifications [20] and context-based web services [21], application semantics [22], and test scripts [23]. Although the XML notation is efficiently adopted in the antecedent research, these XML documents are created manually.

MSC notation is a modeling language that provides textual and graphical formats for modeling the communication sequences between different systems [1]. It is standardized by International Telecommunication Union (ITU) [1]. Using MSC notation is cost effective because the created models can be reused in the future [24]–[26]. MSC notation provides a valuable means for generating protocols [27] or protocol converters [28]. MSC notation can be used for capturing the requirements of developed systems [29], thus increasing the possibility to reuse the existing models. The ability to visualize the communication sequences is a valuable feature in MSC notation and provides a detailed view of the behavior of the communicating systems [30], [31].

Network monitoring is a procedure that can be used to monitor current traffic conditions [32], to monitor multimedia applications [33], to ensure quality of service [34], and to provide a comprehensive view of the networks for administrators [35]. Network monitoring is a resource-intensive procedure [36]. Therefore, the costs of network monitoring may become high [34], [35].

III. THE MODEL CREATION PROCESS WITH TOOL SUPPORT

The purpose of MGtoolV2 is to utilize the network traffic traces stored in Pcap format and to create network traffic models in XML and MSC formats. This process includes the parsing of the Pcap trace file, collecting all necessary features from the data, and transforming the information into the XML and MSC models.

MGtoolV2 creates the XML models by encapsulating the packet-level details. In encapsulation, the XML model is used for modeling the IP addresses, used protocol, ports, and other protocol-related features. The MSC model will encapsulate a network traffic scenario that contains several packets. In the MSC model, the messages sent between communicating systems refer to the corresponding XML models. This integrates the XML models into the MSC model describing a traffic scenario. The use of MGtoolV2 enables the efficient creation of network models that describe the relevant aspects of network activity.

The following sections will describe the details of the model development process and MGtoolV2. Subsection A describes the network traffic flow and the traffic feature spaces. Subsection B discusses the collection of the

relevant features from the traffic traces. The solutions for the presented research problem are presented in subsection C, which describes the model development process, and in subsection D, which describes the technical components of MGtoolV2. The possibility of changing the abstraction level of the output models is discussed in subsection E.

A. DESCRIBING THE NETWORK TRAFFIC FLOW AND TRAFFIC FEATURE SPACES

The network traffic flow is described using the following equation [37]:

$$\varphi(x,t) = \varphi^t(x), \tag{1}$$

where x is an entity in the network traffic flow. In this case, it represents a network packet. The time value of the network traffic flow is given using t. Phi φ represents the flow notation.

When describing the network packet in the network traffic flow using x, it is further described using the Euclidean space notation E [38]. This is represented using the following equation:

$$x = x_1, x_2, \ldots, x_n \in E^N, \tag{2}$$

where N is the number of relevant features collected from the network packet. These features represent the coordinates in the Euclidean space. It is also important to recognize relevant feature groups from the network packet x. This is done by defining the feature groups using the following equation:

$$x = E_1^{N_1}, E_2^{N_2}, \ldots, E_i^{N_n} \in E^N, \tag{3}$$

where the following condition must be fulfilled:

$$N_1 + N_2 + \cdots + N_n = N. \tag{4}$$

To further specify the feature group E_i^n, the following equation describes the Euclidean space of the given feature group:

$$x^i = x_1^i, x_2^i, \ldots, x_n^i \in E_i^n, \tag{5}$$

where n is the number of features used in the feature group i.

Using this approach, it is easier to handle and analyze the feature groups. For example, if only the source and destination IP addresses are the relevant features, then the Euclidean space notation would be as follows:

$$x = x_1, x_2 \in E^2, \tag{6}$$

where the first coordinate would be the source IP address and the second coordinate would be the destination IP address. The selection of relevant features and feature group definitions will be provided in subsection B.

By utilizing the Euclidean space to describe the features of single network packets, it is possible to apply the distance calculations when utilizing the created models. The distance between two distinct coordinates x and y in the Euclidean space is calculated using the following equation [39]:

$$d(x,y) = \|x - y\| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}. \tag{7}$$

These mathematical tools are very efficient when comparing a real network packet against the created model. The tools will be useful especially when the models are utilized in a given application domain, such as in network monitoring. The utilization of the mathematical tools is further discussed in section 4.

B. COLLECTING RELEVANT TRAFFIC FEATURES

Collecting the relevant features is an important procedure in the automatic model creation process. The most obvious features are IP addresses and ports [11]. The packet headers provide many additional features that can be exploited. Frame and payload sizes and the transport layer protocol provide valuable information that can be used in the model creation process [9], [13]. The features are required especially in the XML model creation procedure, to depict a network packet.

The MSC model creation procedure also requires port and IP address features. These features define whether the created MSC model contains details (network activity between specific hosts) for a specific session or for network traffic that is sent between multiple hosts and ports. The duration of traffic flow [9], [13] is the most important collected feature for the MSC model creation procedure, because it indicates the time required by the traffic flow.

Table 1 describes the features that are collected by MGtoolV2. The first feature group contains the IP address as a feature type and it further contains source IP address and destination IP address features. This first feature group also contains the port information by identifying the source and the destination ports. The Euclidean space is defined for the first feature group as follows:

$$x^1 = x_1^1, x_2^1, x_3^1, x_4^1 \in E_1^4. \tag{8}$$

TABLE 1. List of features collected by MGtoolV2.

Feature group	Feature type	Collected feature	Feature notation	Euc. Space
1	IP address	Source IP address	x_1^1	
		Destination IP address	x_2^1	
			x_3^1	
	Port	Source port	x_4^1	E_1^4
		Destination port		
2	MAC address	Source MAC address	x_1^2	
		Destination MAC address	x_2^2	E_2^2
3	Protocol	Network layer protocol (IP, ARP, etc.)	x_1^3	
		Transport layer protocol (TCP, UDP, IMCP, etc.)	x_2^3	E_3^2
4	Protocol flags	IP fragmentation flag	x_1^4	
		IP fragmentation offset	x_2^4	
		TCP flags (SYN, ACK, FIN, etc.)	x_3^4	E_4^5
		IMCP type	x_4^4	
		ICMP code	x_5^4	
5	Size	Frame size	x_1^5	
		Payload size	x_2^5	E_5^2
6	Session	Duration	t	
		Binding	true/false	

The second feature group is the MAC (Media Access Control) address, containing the source MAC address and the destination MAC address features. The MAC address feature can be utilized for modeling network interface numbers. The Euclidean space is defined for the second feature group as follows:

$$x^2 = x_1^2, x_2^2 \in E_2^2. \tag{9}$$

The third feature group contains the protocol features. The features collected by MGtoolV2 are the network layer protocol, such as IP and ARP (Address Resolution Protocol), and the transport layer protocol, such as TCP and UDP. The Euclidean space is defined for the third feature group as follows:

$$x^3 = x_1^3, x_2^3 \in E_3^2. \tag{10}$$

The fourth feature group contains the protocol-related flags. This group contains the IP fragmentation flag and the fragmentation offset features, the TCP flags feature, and the ICMP type and the ICMP code features. The group containing the protocol-related flag features can be further increased to cover more protocols and features when this is required by the application domain. The Euclidean space is defined for the fourth feature group as follows:

$$x^4 = x_1^4, x_2^4, x_3^4, x_4^4, x_5^4 \in E_4^5. \tag{11}$$

The fifth feature group is dedicated to the size of the frame. This group contains the features for the frame size and the payload size. In this case, the frame size feature indicates the total size of the network packet. The payload size indicates the size of the payload contained by the network packet. The Euclidean space is defined for the fifth feature group as follows:

$$x^5 = x_1^5, x_2^5 \in E_5^2. \tag{12}$$

The following equation represents the Euclidean space for the complete network packet, which is divided into the five different feature groups:

$$X = E_1^4, E_2^2, E_3^2, E_4^5, E_5^2 \in E^{15}, \tag{13}$$

where condition from Eq. 4 is fulfilled:

$$4 + 2 + 2 + 5 + 2 = 15. \tag{14}$$

Feature group six contains the session features for a network traffic flow. The group contains the duration feature for defining the duration (in seconds) of the modeled traffic flow. To capture this feature, the value t from Eq. 1 is used. The feature called binding is defined if the traffic flow is transmitted between two different IP addresses and ports. For example, an FTP (File Transfer Protocol) session between two different hosts can be bound. The binding is set as true if the following conditions are fulfilled:

$$x \in E_1^4, y \in E_1^4,\ x \neq y,\ \nexists z \in E_1^4, \tag{15}$$

where x and y are used in the binding and there is no other point z that would be related in the same traffic flow.

Feature groups one to five are utilized in the XML model creation procedure. These features are transformed into the XML model using the specified XML elements, as presented in Table 1. Feature group six is utilized in the MSC model creation procedure. Feature group one is also further utilized, especially when defining the binding feature for the MSC model between different IP addresses (sources and destinations) and ports.

Fig. 1 depicts how different features are transformed into the XML and MSC models. The MSC model is depicted in the upper box in which the graphical and textual formats are provided. It describes a simple ICMP request-reply traffic flow. Both messages are defined in detail using the XML models. In the MSC model, the binding feature is defined using the 'BIND=ALL' section. The messages referring to the XML model are defined using arrows with message descriptions. The duration feature is depicted using 'Timer start T(1)' and 'Timer stop T', where a value of 1 indicates that the traffic flow is conducted within one second. When representing the ICMP request message using x_1 and the ICMP reply message using x_2, the flow can be presented using Eq. 1:

$$\varphi(x_1, 0) = \varphi^0(x_1), \quad \varphi(x_2, 1) = \varphi^1(x_2). \tag{16}$$

The ICMP request message is depicted in the middle box, and the ICMP reply message is depicted in the lower box

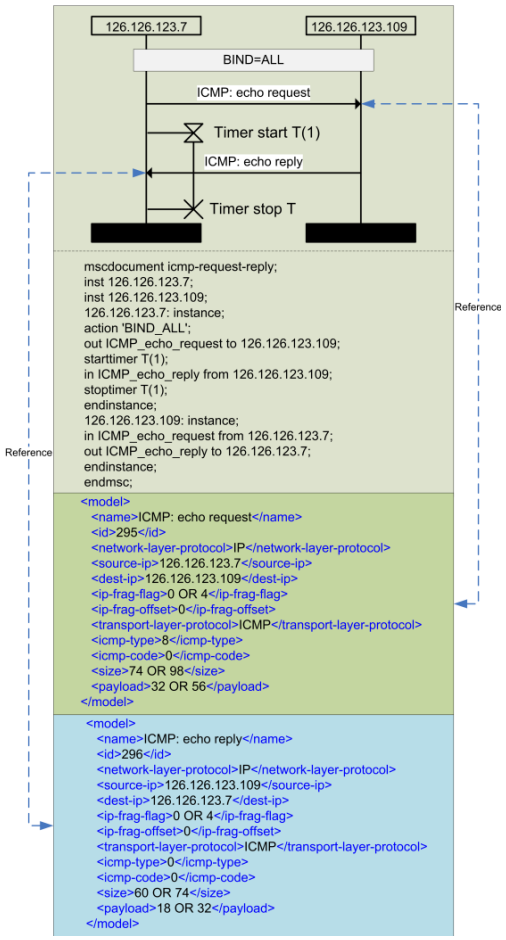

FIGURE 1. Features transformed into the XML and the MSC models.

in Fig. 1. The transformation of the features from Table 1 into the XML model elements in Fig. 1 is self-explanatory. An 'OR' operator can be utilized to indicate possible values for the corresponding feature. For example, the size of the ICMP request frame is either 74 or 98 bytes. MAC address features are omitted for simplicity.

C. THE PROPOSED PROCESS FOR CREATING THE MODELS

The model creation process is depicted in Fig. 2. The predefined process enables the automation of the model creation where each step is executed automatically. The Pcap traffic traces are required, for use in the model creation process. These Pcap traces can be recorded using a tool such as Wireshark[2]. Many network traffic researchers also have a vast number of different traffic traces that can be directly utilized in this model creation process.

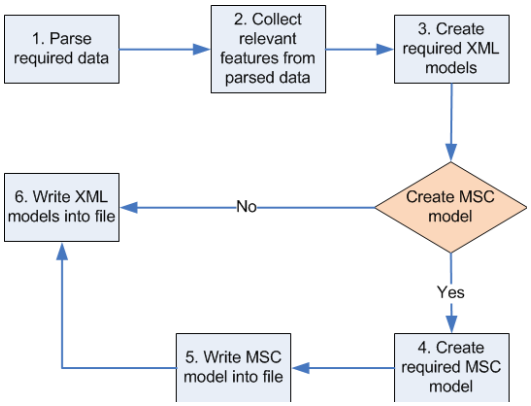

FIGURE 2. The process for automatic model creation.

During the first step depicted in Fig. 2, the Pcap data parser component parses the Pcap trace file and gathers the data for the feature selection step. During this parsing step, the Pcap data parser reads the data from the Pcap trace file using the WinPcap[3] (the industry standard Windows packet capture library) interface. When the Pcap trace file is parsed, MGtoolV2 proceeds to the next step.

The second step is dedicated to collecting the relevant features from the parsed data. During this step, the features are collected according to the choices made by the user. When the abstraction level is elevated, some of the information is filtered out. For example, the IP address and port information are discarded and only the protocol-specific features are collected.

In the third step, the XML models are created. This step is straightforward. MGtoolV2 creates the XML models according to the collected features. When the XML models are created and MSC model creation is enabled, the model creation process proceeds to step four. Otherwise, the process continues from step six.

[2]http://www.wireshark.org/
[3]http://www.winpcap.org/

The fourth step contains the procedure that is responsible for creating the MSC model. The MSC model describes an activity that contains more than just one network packet. In this step, MGtoolV2 utilizes the previously created XML models. A message in the MSC model contains a reference to the corresponding XML model, which further describes the packet-level details of the message. This procedure ties the XML models into the MSC model and specifies the duration of the activity flow.

Steps five and six contain the writing procedures. In step five, the MSC model is written into a file using basic file-writing procedures. Step six contains procedures for writing the XML models into a file.

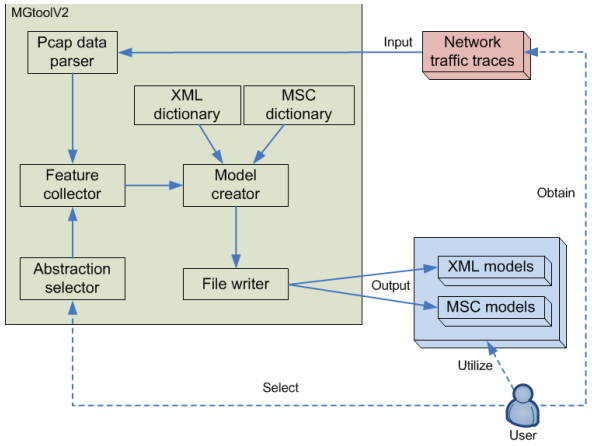

FIGURE 3. The components of MGtoolV2.

D. TOOL SUPPORT FOR AUTOMATING THE MODEL CREATION PROCESS

Fig. 3 depicts the components of MGtoolV2 and the role of the user. The process depicted in Fig. 2 is mapped into functional components of MGtoolV2, as depicted in Fig. 3. MGtoolV2 implements and automates the model creation process. A screenshot from MGtoolV2 is depicted in Fig. 4.

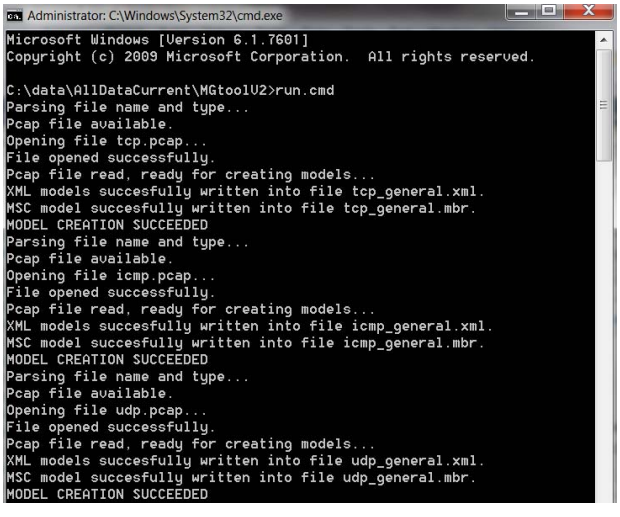

FIGURE 4. Screenshot from MGtoolV2.

The role of the user is to obtain the Pcap traces that are used in the model creation process. The Pcap traces are the inputs required by MGtoolV2. As an output, MGtoolV2 will create the XML and MSC models, which will be further utilized by the user. The application domains in which the models can be utilized are network monitoring, protocol conformance testing, and network analysis. MGtoolV2 also provides a possibility to choose the abstraction level for the created models. The user selects whether the created models are very detailed models or abstract models, and MGtoolV2 creates the models accordingly and automatically. Therefore, the only tasks required from the user are the selection of the abstraction level and obtaining the Pcap trace files.

The Pcap data parser component implements the step one from the process, as depicted in Fig. 2. The component is responsible for reading the network traces that are stored in Pcap format. The network traces serve as an input to MGtoolV2. The Pcap data parser parses and prepares the data for the feature collector component. As the operational environment of MGtoolV2 is Windows, WinPcap is utilized as an interface for reading the data from the Pcap trace file.

The feature collector component is required for collecting the relevant features from the data received from the Pcap data parser component. This component implements the step two from Fig. 2. The feature collector component is able to create data structures for the model creator component. These data structures contain the features described in Table 1, which are required for creating the XML and MSC models. The unnecessary information is filtered out. The collected features were further described in section B.

The abstraction selector component further implements the step two from Fig. 2 and provides the information on which features are collected by the feature collector component. The abstraction selector component will forward this information to the feature collector component. According to the information provided by the abstraction selector, the feature collector component will filter all unnecessary information, and only relevant features are forwarded to the model creator component.

The model creator component utilizes two different components in the model creation process. The first component is the XML dictionary component. This component provides, in a textual format, the elements that are used within the XML model. The second component, called the MSC dictionary component, contains the MSC keywords that are used in creating the textual MSC models. The standard of the MSC notation is extensive [1]. For further information, the MSC keywords and the textual notation details can be found from the MSC standard [1]. The model creator will create the XML and MSC models according to the data structures received from the feature selector component. The model creator component transforms the features collected from the Pcap traces into meaningful XML and MSC models. These three components implement the steps three and four from Fig. 2.

The file writer component is responsible for writing the XML and MSC models created by the model creator component into corresponding files, thus implementing the steps five and six from Fig. 2. The XML and MSC models are the output received from MGtoolV2. Fig. 1 depicts the concrete output formats for the XML and MSC models. The function of the file writer component is based on basic file-writing procedures, and the XML and MSC models are written into the files using textual format. The textual format is used because then the models are machine readable. This makes the models usable in different application domains.

E. RAISING THE ABSTRACTION LEVEL OF THE CREATED MODELS

MGtoolV2 is able to create the XML and MSC models using two different abstraction levels. Fig. 5 depicts two different XML models from the same TCP packet, which has 'PSH-ACK' flags set. The XML model in the upper box describes an abstract model for the corresponding packet. It describes the IP addresses and the ports using 'any' operator, which means that the IP address or the port can be anything. The size of the frame and the payload are indicated using a space operator. The space '60-281' means that the frame size can be anything between 60 and 281 bytes.

The second level contains very specific details. At this level, all features are transformed into the XML model with specific details. This is depicted in the lower box in Fig. 5. MAC addresses, IP addresses, and ports are defined specifically using the 'OR' operator. Different frame and payload

```
<model>
  <name>General TCP packet: PSH ACK</name>
  <id>20002</id>
  <network-layer-protocol>IP</network-layer-protocol>
  <source-ip>any</source-ip>
  <dest-ip>any</dest-ip>
  <ip-frag-flag>4</ip-frag-flag>
  <ip-frag-offset>0</ip-frag-offset>
  <transport-layer-protocol>TCP</transport-layer-protocol>
  <dest-port>any</dest-port>
  <source-port>any</source-port>
  <tcp-flags>24</tcp-flags>
  <size>60-281</size>
  <payload>6-227</payload>
</model>
```

```
<model>
  <name>Specific TCP packet: PSH ACK</name>
  <id>20001</id>
  <source-mac>00:16:47:66:6a:43 OR 00:1e:4f:f4:8a:3e</source-mac>
  <dest-mac>00:1e:4f:f4:8a:3e OR 00:16:47:66:6a:43</dest-mac>
  <network-layer-protocol>IP</network-layer-protocol>
  <source-ip>130.188.4.52 OR 130.188.93.128</source-ip>
  <dest-ip>130.188.93.128 OR 130.188.4.52</dest-ip>
  <ip-frag-flag>4</ip-frag-flag>
  <ip-frag-offset>0</ip-frag-offset>
  <transport-layer-protocol>TCP</transport-layer-protocol>
  <dest-port>13466 OR 21</dest-port>
  <source-port>21 OR 13466</source-port>
  <tcp-flags>24</tcp-flags>
  <size>
    124 OR 66 OR 91 OR 67 OR 109 OR
    92 OR 60 OR 73 OR 281 OR 70 OR
    68 OR 82 OR 59 OR 71 OR 62 OR
    84 OR 63 OR 107 OR 78 OR 98 OR
    123 OR 79
  </size>
  <payload>
    70 OR 12 OR 37 OR 13 OR 55 OR
    38 OR 6 OR 19 OR 227 OR 16 OR
    14 OR 28 OR 17 OR 30 OR 53 OR
    24 OR 44 OR 69 OR 25
  </payload>
</model>
```

FIGURE 5. XML models describing different abstraction levels.

sizes are listed using the 'OR' operator. This differs from the space operator. The listing allows only specific values, whereas the space gives a range between two values.

The abstraction levels are also available for the MSC models. The MSC model depicted in Fig. 1 is a specific model and the flow is described in Eq. 16. The model defines the binding feature and the duration feature. With these features, the MSC model describes a specific traffic flow between two specific IP addresses within a one-second time interval. The second level of the abstraction is depicted in Fig. 6. In the more abstract MSC model, the duration and the binding features are omitted. This means that the MSC model depicted in Fig. 6 is an abstract model that describes a general ICMP request-reply traffic flow with no IP address and duration restrictions. The flow is described using the following equation:

$$\varphi\,(x_1, t_1) = \varphi^{t_1}\,(x_1), \quad \varphi\,(x_2, t_2) = \varphi^{t_2}\,(x_2), \tag{17}$$

where t_1 and t_2 can contain arbitrary values fulfilling the condition $t_1 < t_2$.

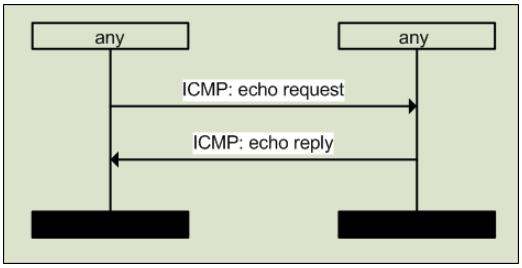

FIGURE 6. An MSC model using higher abstraction level.

The selection of the abstraction level influences the Euclidean space of the feature groups and how they are handled during the model creation process. For example, the different frame and payload sizes listed in the XML model depicted in the lower box of Fig. 5 indicate that there are 22 different values available for the frame size feature and 19 different values for the payload size feature. As the payload size is proportional to the frame size, there are overall 22 coordinates (frame size - payload size pairs) available in the Euclidean space E_5^2. When the abstraction level is raised in the creation of abstract models, then the possible coordinate values for the given Euclidean space will increase. For example, the XML model depicted in the upper box in Fig. 5 indicates that the frame size of the modeled network packet can have values between 60 and 281. This means that there are 221 coordinates (frame size - payload size pairs) available in the Euclidean space E_5^2. If the feature is defined using the 'any' operator, then it means that the whole Euclidean space for the given feature is available.

IV. EXPERIMENTAL CASE: CREATING MODELS FOR NETWORK MONITORING

As an experimental case, MGtoolV2 is utilized to create models automatically for network monitoring. In this case, the XML and MSC models are utilized for describing the network traffic. During the model creation process, MGtoolV2 is utilized for automating the model creation procedures. This will decrease remarkably the manual effort required in the model creation process.

The following subsection describes the experiments made in automatically creating the XML and MSC models according to the given network traffic traces. After the automatic creation of the model sets, we describe the theoretical utilization of the created model sets and provide examples of comparing the traffic flow against the created models.

A. CREATING THE MODEL SETS AUTOMATICALLY

Suitable Pcap trace files are needed in the model creation process. These files included FTP transactions between the client and the server, ICMP, UDP, and TCP traffic. In addition, a trace containing VRRP (Virtual Routing Redundancy Protocol) traffic and a trace containing a mix of TCP, UDP, and ICMP packets were used during the model creation process. The seventh trace contained mixed UDP and TCP traffic in which the packets were sent from the same source IP address. Table 2 describes the detailed contents of each Pcap trace file.

TABLE 2. Pcap trace files used in the experiments.

Trace name	Number of packets	Duration (seconds)	Number of communicating systems
FTP	32	30	2
TCP	11	1	2
UDP	82	92	36
ICMP	18	1529	11
VRRP	31	89	2
Mixed	22588	1720	327
One source	66	481	1 source, 7 destinations

Table 3 lists the created XML and MSC models for each trace, using different abstraction levels (either specific or generic). Each trace file, except the trace file containing a large set of mixed UDP and TCP traffic, was utilized four times in the model creation process. The first two executions concentrated on creating only the XML models. The first execution created the abstract XML models according to the given Pcap trace file. For example, MGtool created one abstract XML model according to the FTP trace file and five abstract XML models according to the TCP trace file. The second execution created specific XML models. As an example, 32 XML models were created according to the same FTP trace file and 11 XML models according to the TCP trace file. The last two executions also created MSC models. The first execution created the abstract MSC model with corresponding abstract XML models, and the second execution created the specific MSC model with the specific XML models. For example, MGtoolV2 created one abstract MSC model with two abstract XML models from an ICMP trace file. When the specific models were created, MGtoolV2 created one specific MSC model

TABLE 3. List of the models created by GtoolV2.

Number	Trace name	Abstraction	Created XML models	Created MSC model
1	FTP	General XML	1	No
2	FTP	Specific XML	32	No
3	FTP	General MSC	1	Yes
4	FTP	Specific MSC	32	Yes
5	TCP	General XML	5	No
6	TCP	Specific XML	11	No
7	TCP	General MSC	5	Yes
8	TCP	Specific MSC	11	Yes
9	UDP	General XML	1	No
10	UDP	Specific XML	82	No
11	UDP	General MSC	1	Yes
12	UDP	Specific MSC	82	Yes
13	ICMP	General XML	2	No
14	ICMP	Specific XML	18	No
15	ICMP	General MSC	2	Yes
16	ICMP	Specific MSC	18	Yes
17	VRRP	General XML	1	No
18	VRRP	Specific XML	31	No
19	VRRP	General MSC	1	Yes
20	VRRP	Specific MSC	31	Yes
21	One source	General XML	5	No
22	One source	Specific XML	66	No
23	One source	General MSC	5	Yes
24	One source	Specific MSC	66	Yes
25	Mixed	General XML	11	No
26	Mixed	Specific XML	22588	No
Overall models created			23109	12

with 18 specific XML models for the same ICMP trace file.

Raising the abstraction level decreases the number of created models, because similar features can be unified into a single model. For example, the UDP trace file contains only UDP packets, and therefore MGtool is able to unify the features related to UDP and creates only one abstract XML model. The abstract MSC model created according to the UDP trace file contains an equal number of messages and packets in the UDP trace file. However, the duration and binding features are omitted, and each message in the MSC model refers to the same abstract XML model.

When a set of specific models is required, the number of created XML models is equal to the number of packets contained in the trace file. This means that a unique XML model is created for each packet contained in the trace file. For example, MGtoolV2 created 22588 XML models when the trace file contained a mix of TCP and UDP packets. The specific MSC model contains the duration and binding features, and a message is created for each network packet in the trace file. However, in the specific MSC model, each message refers to a corresponding specific XML model. For example, the specific MSC model created according to the VRRP trace file contains 31 messages. Each message refers to a unique XML model, so that 31 specific XML models are also created.

As described in Table 3, MGtool was used to create 26 different sets of models from the seven different Pcap trace files. MGtoolV2 created a total of 23109 XML models and 12 MSC models. The biggest set of models was created according to a mixed set of UDP and TCP packets; MGtoolV2 created 22588 XML models. The time that was used by MGtoolV2 to create all 26 sets of XML and MSC models was 15 seconds.

B. APPLYING THE EQUATIONS TO SUPPORT THE UTILIZATION OF THE MODEL SETS

The equations described in section 3 provide powerful tools during the utilization of the created models. As an example, the ICMP Pcap trace file provided 18 different network packets describing either an ICMP request message or an ICMP reply message. Assuming that MGtoolV2 created two distinct models for these two ICMP messages, as described on the 13th row in Table 3, it is possible to utilize the models to analyze real network packets. The Euclidean distance equation (Eq. 7) can be utilized to compare the model with a real network packet by calculating the distance between the model and the real network packet.

The two distinct models for the ICMP request (I_{rq}) and the ICMP reply (I_{rp}) messages can be represented using the following notation, respectively:

$$I_{rq} \in E^{15}, \tag{18}$$

$$I_{rp} \in E^{15}, \tag{19}$$

where $I_{rq} \neq I_{rp}$. A single network packet in the network traffic flow at time t can be represented using the flow equation (Eq. 1):

$$\varphi(X, t) = \varphi^t(X),\ X \in E^{15}. \tag{20}$$

Utilizing the Euclidean distance equation (Eq. 7), the distance between the model I and the real network X is calculated as follows:

$$d(I, X) = \|I - X\| = \sqrt{\sum_{i=1}^{15} (I_i - X_i)^2}. \tag{21}$$

If $d\left(I_{rq}, X\right) = 0$, then the real network packet X at time t is considered as the ICMP request message. On the other hand, if $d\left(I_{rp}, X\right) = 0$, then the real network packet X at time t is considered as the ICMP reply message. If $d\left(I_{rq}, X\right) \neq 0$ and $d\left(I_{rp}, X\right) \neq 0$, then the real network packet X is not considered as either the ICMP request or the ICMP reply. In general, the following definition describes the situation where a real network X at time t is not located in the models:

$$\nexists d(I_x, X) = 0, \tag{22}$$

where I_x is an arbitrary model selected for the distance calculation.

Since the Euclidean space E^{15} is divided into five different feature groups in which each group has its own Euclidean space, it is possible to omit irrelevant feature groups and concentrate on the relevant feature groups. In the case of the ICMP request and reply messages, it is possible to concentrate only on the feature groups defining the protocol information and the protocol-related flag information. These feature groups are defined in Table 1, where the Euclidean

space for the protocol feature group is denoted by E_3^2 and the protocol-related flag group is denoted by E_4^5. When the real network packet X at time t is divided into corresponding feature groups, the distance calculations require less effort. The following equations calculate the Euclidean distance using only the calculations required in the protocol feature group and protocol-related flag feature group, respectively:

$$d(I^3, X^3) = \|I^3 - X^3\| = \sqrt{\sum_{i=1}^{2} (I_i^3 - X_i^3)^2}, \quad (23)$$

$$d(I^4, X^4) = \|I^4 - X^4\| = \sqrt{\sum_{i=1}^{5} (I_i^4 - X_i^4)^2}. \quad (24)$$

The utilization process of the models is easy because the mathematical tools are available. Depending on the application domain, one can select the relevant feature groups for the analysis if the complete Euclidean space is not required. In addition, the utilization of the feature groups enables the usage of different thresholds for the Euclidean distance calculations. Again, depending on the application domain, one can select very strict thresholds for the feature group E_3^2, whereas thresholds for other feature groups can be less strict or may be omitted.

V. EVALUATION

The experiments executed with MGtoolV2 during the model creation process indicated that the proposed tool provides many advantages for creating the XML and MCS models. The following subchapters will describe the evaluation of our proposed work.

A. EVALUATION OF THE PROCESS AND THE TOOL

Utilizing the existing network traffic traces and being able to transform the features of these traces into meaningful XML and MSC models provides a solution for the research problem described in this paper. When the network traffic flow is described using the flow equation (Eq. 1) and the collected traffic features are described using the Euclidean space (Eq. 2), the set of powerful mathematical tools is also available during the utilization of the models.

The research conducted in [2] is enhanced by representing the model creation process. This work also represents the mathematical tools that are omitted from [2]. The combination of the novelty of this work and work in [2] would give very powerful tools for modeling network activities in the field of network monitoring.

MGtoolV2 provides tool support for automating the analysis and collection of network traffic features. As the experiments indicated, MGtoolV2 creates a large set of XML and MSC models automatically (overall 23109 models) and time and cost effectively (within 15 seconds). Automating the model creation process decreases the required manual effort effectively and reduces model development costs. Using MGtoolV2 during the model creation process also minimizes the typing errors and modeling misunderstandings that might occur in manual model development. This further strengthens the solution for the presented research problem. As stated by Voicu et al. [36], network monitoring is a resource-intensive task in many cases. Since the XML and MSC models are created automatically, the high monitoring costs stated by Kuwabara et al. [34] and Schultz et al. [35] can be reduced.

MGtoolV2 is able to create XML and MSC models that describe either very specific details or that are considered as abstract models. With these two abstraction levels, MGtoolV2 is suitable for different application domains in which different levels of details are modeled. The feature collector depicted in Fig. 3 is an important component of MGtoolV2. Depending on the application domain, it can be updated to collect new types of features from the data provided by the Pcap data parser component. For example, if MGtoolV2 was used as part of conformance testing, the feature collector could be updated to collect application layer features.

Pcap traces give a lot of valuable information, and this can be effectively utilized in the model creation process. MGtoolV2 has the ability to unify similar features into a single model, as described in the experiments. It is also considered as an advantage, because unification of traffic features is essential when creating abstract models. This was especially indicated when the trace file containing the mixed set of UDP and TCP traffic was transformed into XML models. The 22588 network packets were transformed and unified into 11 XML models. Machine-readable XML and MSC models can be easily utilized because developing the parser technologies for the XML and MSC formats is trivial. MGtoolV2 supports the model creation process by outputting XML and MSC models that are both descriptive and simple. Utilizing XML and MSC notations enables the possibility to visualize the models. Therefore, these XML and MSC models are not only containers to network traffic features.

Finally, the theoretical utilization of the models provides valuable information on how the mathematical tools can used in the network monitoring domain. The relevant feature groups can be selected for the calculations, and omitting irrelevant feature groups will reduce the required calculation procedures. This makes the proposed MGtoolV2 and the model creation process very useful for different application domains.

B. COMPARISON WITH CURRENT STATE-OF-THE-ART

The traffic features contain valuable information for the model creation process. Xu et al. [12] and Fang et al. [14] concentrate on the collection of IP addresses, ports, and protocol information. Shawky et al. [16] introduce frame sizes and time information into the modeling of network activities. Our work combines these research results and takes this further, where MGtoolV2 collects a diverse set of traffic features and these features are transformed into XML and MSC models. Lai et al. [15] describe the flow concept in modeling network activities. In our work, the MCS model concretizes the modeling of network traffic flow and Eq. 1 is used to describe the traffic flow mathematically. Encapsulating the work from

[12], and [14]–[16] with the XML and MSC modeling functionality, our work provides more usable and comprehensive solution for network activity modeling.

The work presented by Mafra et al. [23], Zhao et al. [22], Zhang et al. [21], and Taketa and Hiranaka [20] all concentrates on utilizing XML notation. The common feature of all of their research is that the XML documents are prepared manually by users. Our work concentrates on automating the creation of the XML and MSC models. Therefore, the XML documents are created automatically, and model development costs are reduced significantly. In resource-intensive network monitoring, the automatic creation of the models is a necessity. This enhances the novelty of our work when compared to works presented in [20]–[23].

A clustering method [10], Hidden Markov Models [9], or a statistical approach [15], [16] are good approaches for indicating the network traffic behavior. However, when models for describing traffic sequences are required, these approaches are not enough. The novelty of MGtoolV2 is in the ability to describe the traffic sequences elaborately using MSC models. When this is combined with the XML models describing details of single packets, it provides an efficient way to model and to describe the network traffic behavior. When compared to works in [9], [15], and [16] the models created using XML and MSC notations are more usable. The XML and MSC notations are well-known and the models can be deployed using simple parser technologies.

The work presented by Zhang et al. [17] describes a model markup language focusing on domain-specific modeling. The key factors presented in [17] are also considered in our work. Firstly, the possibility to increase the abstraction level of the models created by MGtoolV2 covers the high-level data representation factor. Secondly, the extensibility factor is covered by utilizing well-known XML and MSC notations. Thirdly, using XML and MSC notations fulfills the extensibility factor. The MSC notation is standardized by ITU and the XML notation provides formal approach to represent packet-level details. All this combined with the powerful mathematical tools enhances the utilization of the models substantially and is a distinguishing factor from the work presented in [17].

Jukic and Kunstic [11] concentrate on presenting multi-level network models focusing on telecommunication networks. The network model gives a possibility to drill down from higher-level towards lower-level details [11]. However, the work in [11] is missing the possibility to focus on the packet and flow-level details. In our work, we use XML notation to describe the packet-level details and MSC notation focuses on the traffic flow. This is considered as an improvement when compared to the work presented in [11]. Combining our technology with the work in [11], a very efficient multi-level network model could be built.

MSC notation provides a powerful means for describing the traffic flows by visualizing the communication sequences [30], [31]. This, combined with our novel integration of the packet-level details (using the XML notation) into the automated modeling process, means that the network traffic flows can be modeled in a more precise way, which requires less manual effort. The utilization of MSC notation is presented in the research of Abdallah and Jard [27], concentrating on automatic protocol generation, and Roychoudhury et al. [28], where the focus is in the protocol conversion. Both of these are good examples in which our work could provide useful support. MGtoolV2 is able to create MSC models with packet-level details that are described using XML notation, thus providing input for protocol generation [27] and for protocol conversion [28]. This is one example of how the proposed process and tool could be used in different application domains and is considered as future work.

Understanding the essential traffic properties is important for simulating network traffic [10]. Concentrating on traffic volumes provides valuable information for simulation purposes [10]. The XML and MSC models created by MGtoolV2 are possible to utilize in the traffic simulation domain. The models also contain detailed information about the traffic flows, with detailed packet-level information. Combining traffic volume information with the XML and MSC models describing packet-level and flow-level details would create an efficient network traffic simulator. Therefore, our work is not limited to network monitoring.

VI. CONCLUSION

In this paper, we presented the following research problem: automating the feature collection and analysis of network traffic traces for the model development process. The solution to the research problem included a model development process in which network traffic traces were used to create traffic models. The procedures in the model development process are automated, which will reduce model development costs and minimize errors that may occur in manual model development.

To support the model development process, a tool to automatically create traffic models was proposed. The tool, MGtoolV2, is able to create XML and MSC models according to the Pcap network traffic trace files. MGtoolV2 was used as part of the model creation process, in which the user selects the Pcap traces and the abstraction level for the models to be created, and MGtoolV2 creates the models automatically according to the given Pcap trace files. Besides the textual XML and MSC models, the mathematical equations were provided, for use in the creation of models. This increases the usefulness of the proposed MGtoolV2 and the model development process.

The experiments conducted with MGtoolV2 indicated its capability to create different and large sets of XML and MSC models automatically, thus reducing the model development costs effectively. The created models can be utilized in many application domains, such as network monitoring.

The future development plans include implementing feature collection functions for application-level features. In addition, utilizing MGtoolV2 in conformance testing is under consideration. Another development trend would be to provide online model creation functions in which MGtoolV2

would read packets directly from a live network and create XML and MSC models accordingly online.

REFERENCES

[1] *Message Sequence Charts (MSC)*, ITU-T Recommendation Z120, Geneva, Switzerland, 2004.
[2] M. Määttä and T. Räty, "Automatic creation of models for network intrusion detection," in *Proc. Comput., Commun. Appl. Conf.*, Jan. 2012, pp. 231–237.
[3] A. Nogueira, P. Salvador, and R. Valadas, "Modeling network traffic with multifractal behavior," in *Proc. 10th ICT*, Mar. 2003, pp. 1071–1077.
[4] V. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Commun. Mag.*, vol. 32, no. 3, pp. 70–81, Mar. 1994.
[5] M. Laner, P. Svoboda, and M. Rupp, "Parsimonious network traffic modeling by transformed ARMA models," *IEEE Access*, vol. 2, pp. 40–55, Jan. 2014.
[6] S. Chinappen-Rimer and G. P. Hancke, "An XML model for use across heterogeneous client-server applications," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 10, pp. 2128–2135, Apr. 2008.
[7] Y. Zhang, "Test-driven modeling for model-driven development," *IEEE Softw.*, vol. 21, no. 5, pp. 80–86, Sep./Oct. 2004.
[8] F. Hernandez-Campos, K. Jeffay, and F. D. Smith, "Modeling and generating TCP application workloads," in *Proc. 4th Int. Conf. Broadband Commun., Netw. Syst.*, Raleigh, NC, USA, Sep. 2007, pp. 280–289.
[9] E. Costamagna, L. Favalli, and F. Tarantola, "Characterization and modeling of campus-level IP network traffic," in *Proc. ICME*, Baltimore, MD, USA, Jul. 2003, pp. 501–504.
[10] K. Fukuda, "Towards modeling of traffic demand of node in large scale network," in *Proc. IEEE ICC*, Beijing, China, May 2008, pp. 214–218.
[11] O. Jukic and M. Kunstic, "Integrated view on telecommunication network status," in *Proc. 34th Int. Conv. MIPRO*, Opatija, Croatia, May 2011, pp. 429–433.
[12] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Internet traffic behavior profiling for network security monitoring," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1241–1252, Dec. 2008.
[13] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Trans. Netw. Service Manag.*, vol. 6, no. 2, pp. 110–121, Jun. 2009.
[14] G. Fang, Z. Deng, and H. Ma, "Network traffic monitoring based on mining frequent patterns," in *Proc. 6th Int. Conf. FSKD*, Tianjin, China, Aug. 2009, pp. 571–575.
[15] Z. Lai, A. Galis, M. Rio, and C. Todd, "Towards automatic traffic classification," in *Proc. 3rd ICNS*, Athens, Greece, Jun. 2007, pp. 19–28.
[16] A. Shawky, H. Bergheim, O. Ragnarsson, A. Wratny, and J. Pedersen, "Characterization and modeling of network traffic," in *Proc. Int. Comput. Eng. Conf.*, Giza, Egypt, Dec. 2010, pp. 72–76.
[17] C. Zhang, A. Bakshi, and V. K. Prasanna, "ModelML: A markup language for automatic model synthesis," in *Proc. IEEE Int. Conf. IRI*, Las Vegas, IL, USA, Aug. 2007, pp. 317–322.
[18] M. Nieminen and T. Räty, "Representing user definable rules for decision making in the single location surveillance point," in *Proc. 3rd Int. Conf. RCIS*, Apr. 2009, pp. 113–120.
[19] P. A. Fishwick, "Using XML for simulation modeling," in *Proc. WSC*, San Diego, CA, USA, Dec. 2002, pp. 616–622.
[20] T. Taketa and Y. Hiranaka, "Network design assistant system based on network description language," in *Proc. 15th ICACT*, Jan. 2013, pp. 515–518.
[21] X. Zhang, H. Liu, and A. Abraham, "A novel process network model for interacting context-aware web services," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 344–357, Jul./Sep. 2013.
[22] Y. Zhao, Y. Cao, Y. Chen, M. Zhang, and A. Goyal, "Rake: Semantics assisted network-based tracing framework," *IEEE Trans. Netw. Service Manag.*, vol. 10, no. 1, pp. 3–14, Mar. 2013.
[23] J. J. Mafra, R. Netto Lacerda, M. Daride Gaspar, D. Senna Guimaraes, and C. A. Monteiro Leitao, "Multiprotocol monitor and simulator for conformance and interoperability tests at smart grid equipment," in *Proc. IEEE PES Conf. ISGT LA*, Sao Paolo, Brazil, Apr. 2013, pp. 1–5.
[24] I. S. Chung, H. S. Kim, H. S. Baes, Y. R. Kwon, and B. S. Lee, "Testing of concurrent program based on message sequence charts," in *Proc. PDSE*, Los Angeles, CA, USA, 1999, pp. 72–82.
[25] A. En-Nouaary, "An incremental testing method based on timed message sequence charts," in *Proc. ICCCE*, Kuala Lumpur, Malaysia, May 2008, pp. 1248–1253.
[26] H. Dan and R. M. Hierons, "Conformance testing from message sequence charts," in *Proc. ICST*, Berlin, Germany, Mar. 2011, pp. 279–288.
[27] R. Abdallah and C. Jard, "An experiment in automatic generation of protocols from HMSCs," in *Proc. 11th Annu. Int. Conf. New Technol. Distrib. Syst.*, Paris, France, May 2011, pp. 1–8.
[28] A. Roychoudhury, P. S. Thiagarajan, T.-A. Tran, and V. A. Zvereva, "Automatic generation of protocol converters from scenario-based specification," in *Proc. IEEE Int. RTSS*, Lisbon, Portugal, Dec. 2004, pp. 447–458.
[29] A. Roychoudhury and P. S. Thiagarajan, "Communicating transaction processes," in *Proc. ACSD*, Guimaraes, Portugal, Jun. 2003, pp. 157–166.
[30] M. Bezdeka, O. Bouda, L. Korenciak, M. Madzin, and V. Rehak, "Sequence chart studio," in *Proc. ACSD*, Hamburg, Germany, Jun. 2012, pp. 148–153.
[31] M. Brumbulli and J. Fischer, "Simulation visualization of distributed communication systems," in *Proc. WSC*, Berlin, Germany, Dec. 2012, pp. 1–12.
[32] M.-S. Li and W. Na, "Study of network monitoring theory in switched Ethernet and its countermeasures," in *Proc. ICICIS*, Sep. 2011, pp. 585–588.
[33] B. Karacali and C. M. Kintala, "Scalable network monitoring for multimedia applications in enterprise networks," in *Proc. 13th ICCCN*, Chicago, IL, USA, Oct. 2004, pp. 329–334.
[34] S. Kuwabara, K. Shimizu, and M. Maruyama, "Adaptive network monitoring system for large-volume streaming services in multi-domain networks," in *Proc. WTC*, Miyazaki, Japan, Mar. 2012, pp. 1–6.
[35] M. J. Schultz, B. Wun, and P. Crowley, "A passive network appliance for real-time network monitoring," in *Proc. ANCS*, Brooklyn, NY, USA, Oct. 2011, pp. 239–249.
[36] R. Voicu, I. C. Legrand, and C. Dobre, "A monitoring framework for large scale networks," in *Proc. IEEE Int. Conf. ICCP*, Cluj-Napoca, Romania, Aug. 2011, pp. 429–432.
[37] (2014, Feb. 4). *Flow (Continuous-Time Dynamical System)* [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Flow_(continuous-time_dynamical_system)&oldid=26520
[38] E. D. Solomentsev. (2014, Feb. 4). *Euclidean Space, in Encyclopedia of Mathematics* [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Euclidean_space&oldid13577
[39] T. R. Walsh. (2014, Feb. 4). *Euclidean Travelling Salesman, in Encyclopedia of Mathematics* [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Euclidean_travelling_salesman&oldid=14125

MARKO MÄÄTTÄ received the M.Sc. (Tech.) degree from the Department of Electrical and Information Engineering, University of Oulu, Finland, in 2009. His master's thesis concentrated on network intrusion detection. Since 2009, he has been a Research Scientist with the VTT Technical Research Centre of Finland, Oulu. His current research interests include safety and security, machine learning, data fusion, and software testing.

TOMI RÄTY received the Ph.D. degree in information processing science from the University of Oulu, Finland. He is currently a Principal Research Scientist with the VTT Technical Research Centre of Finland, Oulu. His current research interests include data analysis, machine-learning technologies, model-based testing, and software platforms. He is the author or co-author of more than 30 papers published in various conferences and journals, and he has served as a Reviewer for multiple journals and at numerous conferences.