INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
Int. J. Network Mgmt 2015; **25**:355–374
Published online 15 July 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nem.1901

A survey of methods for encrypted traffic classification and analysis

Petr Velan,*† Milan Čermák, Pavel Čeleda, and Martin Drašar

Institute of Computer Science, Masaryk University, Brno, Czech Republic

SUMMARY

With the widespread use of encrypted data transport, network traffic encryption is becoming a standard nowadays. This presents a challenge for traffic measurement, especially for analysis and anomaly detection methods, which are dependent on the type of network traffic. In this paper, we survey existing approaches for classification and analysis of encrypted traffic. First, we describe the most widespread encryption protocols used throughout the Internet. We show that the initiation of an encrypted connection and the protocol structure give away much information for encrypted traffic classification and analysis. Then, we survey payload and feature-based classification methods for encrypted traffic and categorize them using an established taxonomy. The advantage of some of described classification methods is the ability to recognize the encrypted application protocol in addition to the encryption protocol. Finally, we make a comprehensive comparison of the surveyed feature-based classification methods and present their weaknesses and strengths.

Received 10 December 2014; Revised 27 May 2015; Accepted 1 June 2015

1. INTRODUCTION

Network visibility is becoming a necessity in current networks. Security, traffic provisioning and failure detection are the prime reasons to deploy traffic measurement. Yet, measurement has other uses, and new ones are still being discovered. For instance, application performance can be measured using the data from the application layer. Information about certain applications can also be used to detect attacks and intrusions on the application level. In contrast to this, the need for protection of transmitted data and user privacy is rapidly increasing. It is for this reason that the encryption of transmitted data is increasingly used. The ratio of encrypted traffic has recently been rising steeply as common Internet services become protected [1]. This change poses a challenge to currently used methods for traffic measurement, for which the identification and analysis of network traffic become more difficult.

Before any further analysis of encrypted network traffic can be performed, the traffic needs to be identified. Statistical and behaviour-based application identification methods are less affected by encryption than deep packet inspection methods. Therefore, much attention has been given to these methods, which are also considered to be privacy conscious. However, information can even be extracted from encrypted connections, mainly from the session's initiation.

The goal of this survey is to provide a comprehensive overview of methods for classifying and analysing encrypted traffic. To the best of our knowledge, this is the first work that comprehensively summarizes available approaches for encrypted traffic classification and analysis. Although there has been much research on traffic classification in the past decade [2–6], most of the existing surveys do not explicitly consider research that targets encrypted traffic. A recent survey by Cao *et al*. [7] describes the fundamentals of encrypted traffic classification. It briefly introduces recent advances and challenges in

*Correspondence to: Petr Velan, Institute of Computer Science, Masaryk University, Botanická 68a, Brno, Czech Republic.
†E-mail: velan@ics.muni.cz

this field. However, the survey covers only a few methods of traffic classification and does not provide any comparison or assessment of these methods. Moreover, while traffic classification is an important part of network monitoring, there are other methods for analysing encrypted traffic, which need to be taken into consideration.

To achieve this goal, we have produced a survey of methods for classifying and analysing encrypted traffic in journals, conference papers, proceedings of specialized workshops and technical reports. We studied works presented in selected computer science journals, mainly the *Communications Surveys & Tutorials*, *Computer Networks*, *International Journal of Network Management* and *Transactions on Network and Service Management*. We also surveyed international conferences such as IMC, PAM, CISDA, CNSM, IM and NOMS over the period 2005–2014. Other methods were found in references provided by the surveyed papers and in papers that referenced them.

The contribution of our work is fourfold. First, we describe several of the most widely used encryption protocols to show their packet structure and standard behaviour in a network. This information forms the basis of all classification protocols, which use either a specific packet structure or a communication pattern to identify the protocol. Second, we investigate what information is provided by these encryption protocols. Most protocols negotiate encryption algorithms in clear text; these data can be monitored, e.g. to reveal the use of weak ciphers. Third, we describe how the structure of encryption protocols can be used to detect these protocols in a network. Traffic classification algorithms using such information are presented, and we describe several open-source tools that implement these algorithms. Fourth, we provide an extensive survey of behaviour-based methods for encrypted traffic classification. We show that surprisingly detailed information can be obtained using these methods. In specific cases, even the content of the encrypted connection can be established. To describe and categorize the classification methods, we use the taxonomy of Khalife *et al*. [8].

Over the last decade, many statistical and machine learning algorithms have been applied to the problem of traffic classification. However, authors use different methodological datasets to evaluate their methods, and the results are therefore not directly comparable. Most of the methods use supervised or semi-supervised machine learning algorithms to classify flows and even determine the application protocol of the flow. Most methods target encryption protocols such as Secure Shell (SSH), Secure Sockets Layer (SSL)/Transport Layer Security (TLS) and encrypted BitTorrent.

This paper is organized into seven sections. Section 2 explains the principles of several widely used encryption protocols. Section 3 describes what information can be obtained from encrypted traffic using specific knowledge of the encryption protocols. Section 4 describes the traffic classification taxonomy used in this article. Section 5 describes traffic classification methods based on the specific knowledge of encryption protocols. Section 6 surveys papers on the classification of encrypted traffic using statistical and behavioural methods. Finally, Section 7 concludes the paper and provides directions for future research.

2. A DESCRIPTION OF ENCRYPTION PROTOCOLS

This section provides a description of chosen encryption protocols. We selected the most widely used protocols to demonstrate the basic principles and show different approaches for secure data transport. Our choice of encryption protocols was also influenced by protocols that are the most commonly used in research of encrypted traffic classification. We shall describe Internet Protocol Security (IPsec), TLS, SSH, BitTorrent and Skype protocols in this section. All these protocols provide confidentiality using encryption, and they usually offer authentication of communication peers, data integrity, replay protection and non-repudiation. We shall describe the use of these protocols, their fundamental properties, the structure of encrypted packets and the type of exchanged data. The information provided in this section will serve as the basis for the next sections of this paper.

Almost all the presented encryption protocols can be divided into two main phases: the initialization of the connection and the transport of encrypted data. The first phase can be further divided to an initial handshake, an authentication and a shared secret establishment. During the first phase, algorithm capabilities are usually exchanged, communication parties are authenticated and secret keys are established. These keys are then used for encrypting transferred data in the second phase. This general protocol scheme is depicted in Figure 1, which, with minor modifications, can be applied to almost any protocol providing confidential data transfer.

DOI: 10.1002/nem

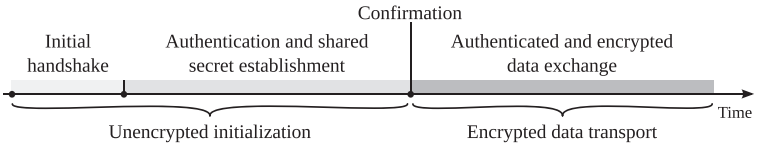

Figure 1. A general scheme of network security protocols

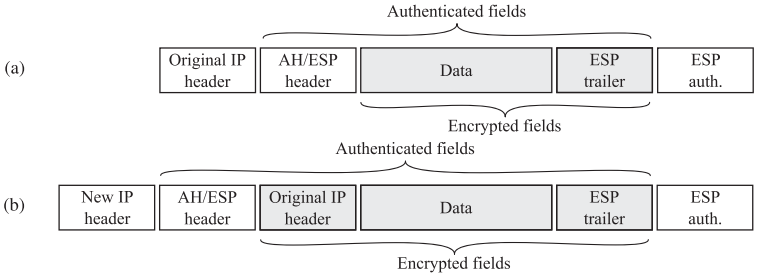

Figure 2. The IPsec packet structure in (a) the transport mode and (b) the tunnel mode

The protocols presented in this section are listed in order of their position in the ISO/OSI reference model [9]. IPsec protocol suite, which operates on the network layer, is described first, followed by TLS and SSH protocols on the presentation layer. BitTorrent and Skype protocols represent the application layer, and they implement their own protocols for secure data transmission.

2.1. Internet protocol security

IPsec is a framework of open standards for ensuring authentication, encryption and data integrity on the network layer. Owing to its location on the network layer, IPsec can protect both the data within the packet and also L3 information (e.g. IP addresses) in each packet [10]. The main advantage of using IPsec is securing the network connection without the necessity of modifying any application on the clients or servers. However, it provides less control and flexibility for protecting specific applications.

IPsec follows the general scheme depicted in Figure 1. The first phase is represented by the Internet Key Exchange Version 2 protocol [11]. IPsec uses a User Datagram Protocol (UDP) protocol on the port 500 through which all messages covering the initial handshake, the authentication and the shared secret establishment run. Two protocols could be used in the second phase of IPsec: Authentication Header (AH) and Encapsulating Security Payload (ESP). In the initial version of IPsec, the ESP protocol providing data confidentiality did not include authentication, so ESP and AH were used together. Nowadays, the current version of ESP contains authentication, and AH has become less significant, although it is still used to authenticate portions of packets that ESP cannot manage.

The ESP protocol is the main protocol of IPsec. It provides data confidentiality, origin authentication, connectionless integrity, an anti-replay service and limited traffic flow confidentiality [12]. ESP adds a header and a trailer to each transferred packet (Figure 2) placed according to the transport mode used. The ESP and AH protocols can operate in two modes: *transport* and *tunnel*. In the *tunnel* mode, a new IP header is created for each packet with endpoints of the tunnel as the source and destination addresses; the original IP header is used in the *transport* mode.

2.2. Transport layer security

TLS [13] is based on the SSL version 3 (SSLv3) protocol [14] and provides transport level security directly on top of the TCP protocol. Specifically, it provides confidentiality, data integrity, non-repudiation, replay protection and authentication through digital certificates. The TLS is currently one of the most common protocols for securing network communication. It is mainly used for securing

DOI: 10.1002/nem

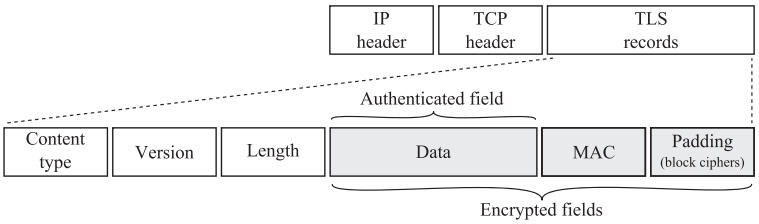

Figure 3. The TLS Record packet format

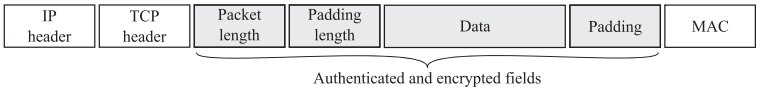

Figure 4. The SSH protocol packet format

HTTP, FTP, Simple Mail Transfer Protocol (SMTP) sessions, as well as for Virtual Private Networks (VPN) or Voice over Internet Protocol (VoIP). The protocol design is layered and consists of different sub-protocols, as well as configurable and replaceable cryptographic algorithms [15].

The main part of TLS is the Record Protocol [13], which acts as an envelope for application data as well as TLS messages. In the case of the application data, the Record Protocol is responsible for dividing the data into optionally compressed fragments. The addition of fragments to the record is complemented by Message Authentication Code (MAC). For more details, see Figure 3. Depending on the selected security algorithms, a fragment and MAC are encrypted together and sent as one TLS record. A packet may contain more than one record to avoid sending multiple short packets.

During the first phase of a TLS connection, communication parties are usually authenticated (more often, we can see only server authentication) using an X.509 certificates chain [16], as shown in the general scheme in Figure 1. Alternatively, a previous connection can be resumed without authentication. TLS messages exchanged during this phase are unencrypted and do not contain MAC until the shared keys are established and confirmed. In the second phase, these keys are used directly by the Record Protocol, which is based on the selected algorithms ensuring communication security.

2.3. Secure shell protocol

In a similar fashion to the TLS protocol, the SSH protocol [17] exists as a separate application on top of TCP. This protocol uses a client–server model where the server usually listens to the TCP port 22. SSH was originally designed to provide a remote login access to replace unsecured Telnet connections. Nowadays, it can be used not only for a remote login and a shell but also to allow file transfers using the associated SSH File Transfer Protocol (SFTP) [18] and a Secure Copy (SCP) [19] protocol, or by VPN. The SSH protocol provides user and server authentication, data confidentiality and integrity and, optionally, compression.

SSH consists of three protocols, of which the most important is the Transport Layer Protocol, which provides the establishment of the whole connection and its management. It defines the SSH packet structure, which is depicted in Figure 4. The MAC is computed on a plaintext payload together with the packet length, the padding and a continuously incremented sequence number not present in the packet itself. Other SSH protocols are the User Authentication Protocol and the Connection Protocol for multiplexing multiple logical communications channels [20].

Each SSH connection passes through the same phases, which were depicted in Figure 1. In the first phase, a TCP connection is established, and information about preferred algorithms is exchanged. During authentication, a server sends its public key, which must be verified by the client (using a certification authority or manually through a different channel). The shared keys are subsequently established and confirmed. All following packets are then encrypted and authenticated.

DOI: 10.1002/nem

2.4. BitTorrent

BitTorrent [21] is an application protocol based on the principle of peer-to-peer network communication for sharing large amounts of data over the Internet. Originally, the protocol did not ensure any type of network communication security. Once the popularity of this protocol increased, some Internet Service Providers (ISP) started to limit this type of traffic. As a response to this, the Message Stream Encryption (MSE) algorithm [22], also known as Protocol Encryption, was introduced. It serves as an obfuscation algorithm to make BitTorrent traffic identification more difficult. In addition to obfuscation, the mechanism also ensures some level of confidentiality and authentication for communicating peers.

The MSE protocol specification [22] describes MSE as a transparent wrapper for bidirectional data streams over TCP, which prevents passive eavesdropping and thus protocol content identification. MSE is also designed to provide limited protection against active man-in-the-middle attacks and port scanning by requiring a weak shared secret to complete the handshake. The major design goal was payload and protocol obfuscation, not peer authentication and data integrity. Thus, it does not offer protection against adversaries that already know the necessary data to establish connections (that is, the IP/port/shared secret/payload protocol).

The first general phase of the MSE protocol follows a TCP three-way handshake and starts with a newly generated Diffie–Hellman (DH) public key exchange (together with random data padding for better obfuscation). The shared key is computed by the DH key and combined with hashed information about the requested data, which acts as a pre-shared secret. The packet's payload is completely encrypted by an RC4 stream cipher after successfully confirming the shared key.

2.5. Skype

Skype [23] is a peer-to-peer based VoIP application providing not only video chat and voice calls but also an instant messaging and file exchange service. As the protocol used is not publicly known, it is not possible to accurately describe its specific details. The main reason for this is network data obfuscation to make the detection of Skype traffic more difficult, which is similar to the BitTorrent protocol.

The Skype protocol operates over both UDP and TCP protocols depending on network characteristics. If the network has no restrictions, the application usually sends data traffic over UDP and the signalling traffic over TCP [24]. If UDP cannot be used (for example, a firewall prevents users from using such a protocol), Skype sends both the signalling and the data traffic over TCP. When TCP is used, the connection is usually established over port 80 or 443, which masks Skype traffic as standard web traffic.

Skype uses the TLS protocol over the TCP port 443 and a proprietary protocol over port 80 [24] for securing and obfuscating generated traffic with each communicating peer. TLS is also used in communication with other Voice over IP solutions, where it is used for protecting Session Initiation Protocol (SIP) messages [25]. Skype uses a proprietary protocol for communication over UDP. To offer a reliable connection, UDP packets contain an unencrypted header with a frame ID number and function mode fields. The encryption in UDP connections is used only for obfuscation and not for confidentiality; therefore, there is no generated shared secret, only a proprietary key expansion algorithm. UDP connections do not follow the general scheme in Figure 1, because encrypted data are directly transferred without an initialization phase.

3. INFORMATION EXTRACTION FROM ENCRYPTED TRAFFIC

Network monitoring is one of the main pillars of network security. If the appropriate data are collected, it is possible to detect network attacks and trace attackers, detect security policy violations and monitor network applications performance. If encrypted traffic is used, the possibility of information extraction is significantly limited. Nevertheless, it is possible to obtain some information from this traffic, primarily from the unencrypted initialization phase, and also from the encrypted transport phase. This section begins with a description of the initialization phase, which is then followed by a description of methods that use traffic feature analysis to gain information from the transport phase.

DOI: 10.1002/nem

3.1. The unencrypted initialization phase

Almost all network protocols ensure secure data transfer by means of encryption containing an unencrypted initialization phase, as depicted in Figure 1. Because the data exchanged at this stage are not encrypted, they can be easily extracted and used for monitoring network traffic. Generally, two types of information common to most protocols can be extracted during this phase. The first type covers the connection itself, and its properties exchanged in the initial handshake. The second type covers communicating peers' identifiers, which are exchanged in the authentication phase.

During the initial handshake, parameters of a connection are negotiated, such are cipher suites and which protocol version is used. This dynamic setting of the connection properties enables backward compatibility for different versions of software or is used to set a different level of security based on established security policies. Some examples of this are data authentication and compression in addition to encryption itself. The list of possible identifications, with references to algorithm specification for IPsec, TLS, SSH and other protocols, can be found in IANA Protocol Registries [26]. All of this information can be used for proper connection characterization and correct parsing of other packets in the rest of the connection.

One interesting use of the information from the initial handshake is presented by client fingerprinting based on the provided cipher suites. A large amount of cipher-suites types exists, which usually are not all implemented by the client's applications. Therefore, each application specifies the supported cipher suites and also prefers using them during the initial handshake. This makes it possible to passively distinguish specific operating systems, web browsers and other applications, together with their versions, based only on the cipher suites that they use. An example of such client fingerprinting, based on the SSL/TLS initial handshake, is presented by applications from SSL Labs [27] and p0f module [28].

The authentication phase represents the second source of information that can be easily extracted from secured network traffic. Unique identifiers of one or both communicating peers, optionally supplemented by additional data about them, are exchanged during this phase. For example, in the authentication phase of the SSH protocol, the server sends its public key to the user. The user must validate the key and verify that the server knows the appropriate private key [17]. As this information is almost unique for each SSH server, it is possible to detect server changes or man-in-the-middle attacks on them by passive network traffic monitoring.

A very good example of extracting information from communication peers is demonstrated by monitoring the authentication phase of the SSL/TLS protocol. The server, and optionally even the client, sends their X.509 certificates [16] to each other to verify their identities in this phase. This certificate contains a public key signed by the certification authority, which is supplemented with information about the peer and issuer. More detailed content of such certificates is shown in Figure 5.

Monitoring passive certificates enables not only the identification of communicating peers but also the ability to check whether these certificates are valid and contains proper security algorithms to fulfil local security policies. SSL/TLS certificate properties were studied by Holz *et al*. [29] who revealed a great number of invalid certificates and some which were shared between a large number of hosts. The work of Holz *et al.* was followed by that of Durumeric *et al*. [30] who mainly focused on assessing

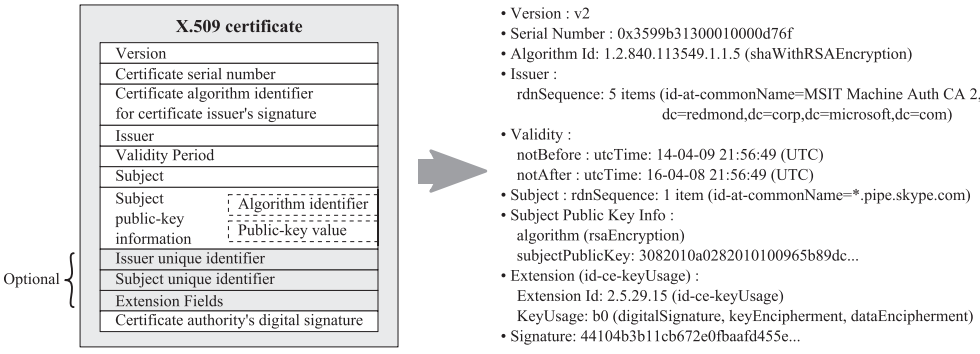

Figure 5. Example of Skype X.509 certificate

DOI: 10.1002/nem

certification authorities. Certificate monitoring can also be used to detect malicious software trying to hide its activities by connecting with their command and controls centres using the SSL/TLS protocol [31].

Even though information extraction from encrypted traffic is not a computationally intensive process, it can provide valuable information. For example, extracting the Server Name Indication [32] can be used by a home router's firewall to filter traffic. The unencrypted initialization phase is often used to recognize encrypted traffic, and the authentication information might be utilized to detect and prevent man-in-the-middle attacks on a network-wide level.

3.2. The encrypted data transport phase

Information about network traffic can be extracted from encrypted data, which is transported between communicating parties. Packets exchanged during the transport phase usually contain only information about the packet itself, such as the length and the authentication field, which are not useful for monitoring network traffic. Nevertheless, two methods do exist to obtain more suitable data.

The first method uses direct traffic decryption, which is possible to perform only if the shared secret of the connection is known. Therefore, decryption can be used in networks on the servers' side where organizations know the private key of the connected server. However, such decryption would be impossible if algorithms were used, which ensure forward secrecy [33].

The second method is based on the extraction of traffic features. An example of such an analysis is presented by Miller *et al*. [34] who monitor the size of TLS encrypted packet sequences. Based on their data, together with various predictive models, they are able to identify a specific web page and deduce its content even though the traffic is encrypted. A similar approach was also used by Koch and Rodosek [35] for analysing SSH traffic. Another example of using traffic features is the work by Hellemons *et al*. [36], which focused on intrusion detection in SSH traffic. Encrypted traffic features could also be used for classifying encrypted traffic, which is described in Section 6.

4. A TAXONOMY FOR TRAFFIC CLASSIFICATION METHODS

The first step in analysing network traffic is identifying the type of traffic measured. Network traffic classification methods are used for this purpose based on the knowledge of the protocol packet structure, communication patterns or a combination of both. The recognition of the TLS protocol, described in Section 2, may be seen as an example of this. This protocol can be identified based on the knowledge of the packet structure, especially the unencrypted packet parts such as the content type, the version and the length. Similarly, the protocol can be identified by analysing its behaviour, e.g. the knowledge of a number and an approximate size of packets sent during the unencrypted initialization phase.

To present the current state of research on encrypted traffic classification in a comprehensive manner, we use a taxonomy of classification methods. We choose the multilevel taxonomy by Khalife *et al*. [8], which provides a detailed categorization of traffic classification methods. This taxonomy is uniquely descriptive and allows us to efficiently categorize all our surveyed classification methods. Figure 6 shows an overview of the taxonomy levels. On the topmost level, the authors distinguish between classification input, technique and output. The input determines the traffic's characteristics, which are used for classification (e.g. packets or flows). The technique describes the core of the classification method, which may be, among others, payload inspection, a statistical method or a machine learning method. The output then describes how the traffic objects (packets or flows) map to traffic classes (application types or application protocols). The traffic classes are of a different granularity. Some methods allow identification of application protocols (e.g. HTTP), and some are more fine grained to detect, for example, a Google search or a Facebook chat.

Tables 1, 2 and 3 provide examples for each of the input, technique and output category. The input and technique tables have the most general categories in the left column, some of which are divided into more specific subcategories. The classification output table is divided horizontally into traffic objects and classes. The objects describe what is being classified, in other words, whether it is each packet, whole flow or a host. Traffic classes describe the type of classification being performed by a specific algorithm.

DOI: 10.1002/nem

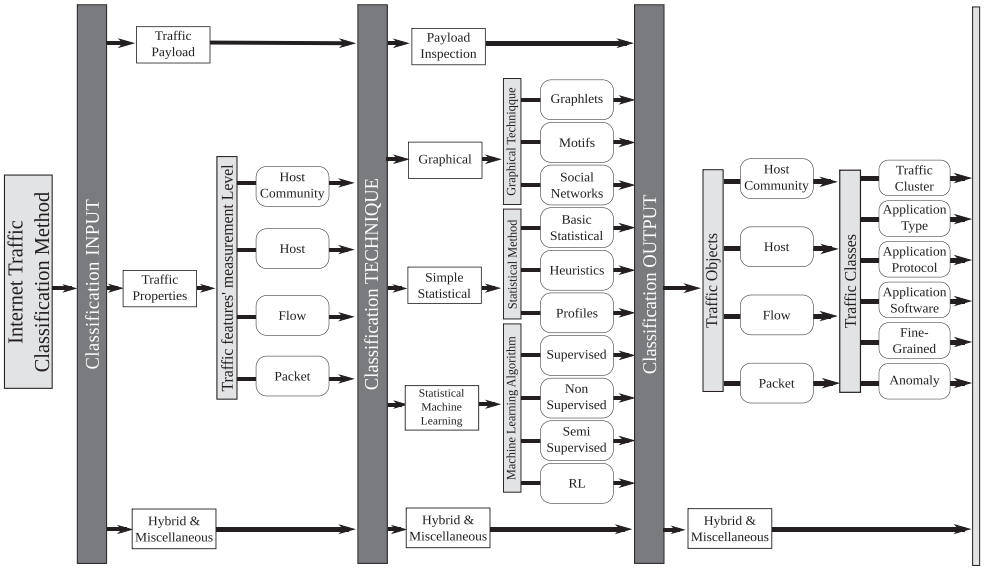

Figure 6. A multilevel taxonomy of traffic classification methods [8]

Table 1. Classification input level

Classification Input		
Traffic payload		Use of application data
Traffic properties (measurement level)	Host community	Graph metrics (diameter, connection degree)
	Host	Number of connections, opened ports
	Flow	Flow size, flow duration
	Packet	Packet sizes, inter-arrival times
Hybrid and miscellaneous		Combination of inputs, external knowledge

Table 2. Classification technique level

Classification technique		
Payload inspection		DPI, examination of first N bytes
Graphical techniques	Graphlets	Relationship between features (ports, addresses)
	Motifs	Patterns of communication
	Social networks	Graph of communication
Statistical method	Basic statistical	Probability density functions of, e.g. packet sizes
	Heuristics	Port-based classification
	Profiles	Host profiling, usage of packet sizes and direction
Machine learning algorithm	Supervised	Hidden Markov models, naive Bayes, k-nearest neighbour, support vector machine
	Non-supervised	Clustering of unlabeled traffic, k-nearest neighbour
	Semi-supervised	Clustering of mixed traffic, k-nearest neighbour
	Reinforcement learning	—
Hybrid and miscellaneous		Combination of methods, external knowledge

DOI: 10.1002/nem

Table 3. Classification output level

Classification output	
Traffic objects	
Host community	Host community is assigned a class, e.g. community of HTTP servers
Host	Host is assigned a class
Flow	Flow is assigned a class
Packets	Packet is assigned a class
Traffic classes	
Traffic cluster	Bulk or small transactions
Application type	Game, browsing, chat
Application protocol	HTTP, HTTPS, FTP
Application software	Client software such as Mail client, FTP client or web browser
Fine grained	Skype voice call, Google search, Facebook chat
Anomaly	Port scan, brute-force attack
Hybrid and miscellaneous	Combination of outputs or classes, external knowledge

5. PAYLOAD-BASED TRAFFIC CLASSIFICATION TECHNIQUES FOR ENCRYPTED TRAFFIC

Almost every network traffic encryption protocol has a specific packet format that differs from others, as was described in Section 2. Thus, with knowledge of these formats, it is possible to distinguish and identify individual protocols by inspecting the packet payload. It is for this purpose that string or regular expression matching algorithms are used with specific protocol patterns. Some examples of contemporary classification tools, which use payload inspection, are discussed in more detail in the first part of this section. The second part presents current research papers that focus on comparing these tools in terms of their performance and success rate.

5.1. Payload-based classification tools

Most network traffic classification tools address all network protocols and not only the encrypted ones. The following examples represent the most widely used tools for classifying network traffic. Most of these tools are also able to distinguish specific network applications, mainly in unencrypted traffic. In terms of the taxonomy, these tools mostly use the *payload inspection* technique on the *traffic payload* classification input to map *flows* to *application protocols*.

PACE [37] is a commercial classification library written in C, which uses pattern matching augmented by heuristics, and behavioural and statistical analyses. In addition to standard protocol and application recognition, it is able to identify obfuscated protocols such as BitTorrent or Skype. According to its web site, PACE is able to identify thousands of network applications and protocols.

Cisco Network Based Application Recognition (NBAR) [38] is another example of a commercial tool for classifying network traffic. This tool is primarily used on Cisco routers for quality and security purposes. According to its authors, NBAR is also able to recognize stateful protocols and non-TCP and non-UDP IP protocols.

nDPI [39] is an open-source classifier forked from the (currently closed) project OpenDPI, which was in turn derived from PACE. nDPI analyses at most eight packets from each connection for classifying traffic; however, each packet is examined separately. If the connection contains multiple matches, then the most detailed match is returned. For encrypted traffic recognition, nDPI contains only an SSL decoder that extracts the host name from the server certificate. Using these names, nDPI is able to identify specific network applications.

Libprotoident [40] is an open-source C library for classifying traffic. In contrast to the previous tools, Libprotoident inspects only the first four bytes of a packet payload for each direction. This makes it much faster but reduces its detection accuracy. The classification uses a combined approach of pattern matching, payload size, port numbers and IP matching.

L7-filter [41] is an open-source classifier for Linux, which is designed to classify traffic on the application layer. The initial phase of classification is based on non-payload data such as port numbers,

DOI: 10.1002/nem

IP protocol numbers and the number of transferred bytes. Payload data are analysed with regular expressions during the second phase. One disadvantage of the l7-filter is that it contains a database with old patterns, which was last updated in 2011.

5.2. A comparison of classification tools

A comparison of the presented open-source tools was introduced by Finsterbusch *et al*. [6]. They prepared a dataset containing the traffic of 14 different network protocols such as DNS, HTTP, BitTorrent and SMTP(S) to compare the tools. Using this dataset, they measured classification accuracy, memory usage, CPU utilization and the number of packets required for proper classification. The comparison showed, amongst other results, that nDPI is not able to classify the BitTorrent protocol with more than a 43% true-positive rate, although it detects SSL/TLS with 100% accuracy. The Libprotoident tool had the highest classification accuracy of the whole analysed traffic, which was able to classify DNS, HTTP, SIP and e-mail protocols with 100% accuracy. Libprotoident also needs the least number of packets on average for classifying real-time traffic. Based on the comparison by Finsterbusch *et al*., we can say that Libprotoident is the most appropriate tool for classifying payload-based traffic, although it is more CPU intensive than the other tools.

Another comparison of the tools was carried out by Bujlow *et al*. [42]. They compared all of the previously presented tools. They prepared a publicly available dataset for the comparison, which contained encrypted and unencrypted network traffic from 17 application protocols, 25 network applications and 34 web services. Their results show that PACE and Libprotoident are the most accurate tools. Nevertheless, the Libprotoident tool was the only classifier able to identify all the encrypted protocols that were tested. Their results were generally very similar to those in the comparison by Finsterbusch *et al*.

In general, the payload-based classification tools use regular expression matching algorithms to identify the encrypted traffic. The main difference between the tools is how much data they need to examine and whether they need both directions of the connection for the classification.

6. FEATURE-BASED TRAFFIC CLASSIFICATION TECHNIQUES FOR ENCRYPTED TRAFFIC

This section surveys feature-based classification methods that specialize in encrypted traffic. These methods do not require any knowledge of the encryption protocol packet structure. Instead, they use specific protocol communication patterns to classify encrypted traffic. These methods are based on the specific protocol differences described in Section 2, such as packet and flow features of unencrypted initialization or the encrypted data transport phase. This approach provides greater generalization and allows these methods to work with new versions and types of encryption protocols without the modification of underlying algorithms.

The taxonomy specified in Section 4 is used to describe the individual methods. Apart from the properties defined by the taxonomy, we also provide information about datasets used for evaluating these methods. This is especially important when additional evaluation is to be performed by other groups to verify the results of the authors. While the taxonomy provides a traffic class for classifying the application protocol, this is sometimes too coarse for our purposes. Most classification methods identify not only the encryption protocol but also the underlying encrypted application protocol. As both cases belong to the application protocol traffic class, we provide further explanation in the description of each classification method.

A slight drawback of flow-based classifiers is in performing classification often after the flow has expired. This prevents the possibility of a real-time response, which has led several research groups to research real-time classification using flow features. Their methods are also included with the others in Table 4 and differentiated by a column describing whether the classification is carried out in real time or not.

Almost 250 discriminators (flow or packet features) are identified by Moore *et al*. [43], which can be used to classify flow records. The authors do not propose any specific classification method themselves; however, most of the classification algorithms use a subset of these discriminators for identifying

DOI: 10.1002/nem

traffic. In terms of the taxonomy, the authors provide a list of traffic features, which are used to infer the category of the traffic's properties for the classification input.

Most of the feature-based traffic classification methods use statistical or machine learning methods. These methods are comprehensively described in [44]. Port-based classification was used in the past to associate applications with network connections, but the accuracy of this method is decreasing with the increased use of dynamic ports and applications evading firewalls. Despite the decreased accuracy, port numbers are often utilized as one of the packet features. Furthermore, port-based classification is still quite often used to establish a ground truth for traffic classification experiments.

For easier orientation, we present the surveyed papers grouped by the class of their traffic classification algorithm. Most of the papers employ supervised machine learning methods (Section 6.1), semi-supervised machine learning methods (Section 6.2) and basic statistical methods (Section 6.3). The rest combine more than one method and are therefore gathered in a hybrid methods category (Section 6.4). For each surveyed paper, we specify the classification technique, classification output and the datasets used in the description of the classification process itself. Table 4 provides a summary of the papers with all of the mentioned properties properly categorized.

6.1. Supervised machine learning methods

Sun *et al*. [45] propose a hybrid method for classifying encrypted traffic. First, the SSL/TLS protocol is recognized using a signature matching method. A naive Bayes machine learning algorithm is then applied to identify the encrypted application protocol. The authors use a combination of public and private datasets to evaluate their method. Background traffic is taken from a public dataset, BitTorrent, eDonkey, HTTP, FTP, Thunder and GRE application protocols. The signature-based recognition of SSL/TLS protocols was tested on HTTPS, TOR, ICQ and other protocols. The identification of the underlying protocol was tested only for TOR and HTTPS protocols.

Okada *et al*. [46] analysed changes in flow features due to encryption. They created a training dataset with HTTP, FTP, SSH, and SMTP application protocols encrypted using Point-to-Point Tunneling Protocol (PPTP) and IPsec tunnels. The authors assessed 49 flow features and analysed which of them are strongly correlated in normal and encrypted traffic. The correlated features were then used to infer functions that transform the features between normal and encrypted traffic. Therefore, standard classifiers can be used to classify the traffic after the transformation. The authors verified their method using several modifications of the naive Bayesian classifier.

Arndt and Zincir-Heywood [47] also concentrated on the classification of encrypted traffic and compared C4.5, k-means and Multi-Objective Genetic Algorithm (MOGA) to this end. The classification of the SSH protocol was used as an example in their study. The authors focused on the accuracy and robustness of the algorithms. Stability was tested using three different public and private datasets for teaching and evaluating the methods. Multiple different flow export settings were tested as well. Altogether, 46 flow features were used in the evaluation; however, a different subset was used by each algorithm. The C4.5 algorithm provided the best robustness, although the MOGA had a very low false-positive rate when used on the same dataset as it was trained on. The C4.5 was recommended for forensic analysis by law enforcement because it is applicable on a variety of networks.

Alshammari and Zincir-Heywood have published several papers [48–53] on traffic classification using various supervised machine learning methods. They focused on recognizing SSH, Skype and, in one case, Gtalk traffic using flow features without port numbers, IP addresses or payloads. A set of 22 flow features was mostly used, although in one case the authors selected the features using genetic programming. Public datasets are used as well as a private dataset generated on a test-bed network. The ground truth for public datasets was gained from port numbers, and the private dataset includes the payload and was labelled with a commercial packet classification tool. The following algorithms for traffic classification were compared: AdaBoost, RIPPER, support vector machine (SVM), naive Bayes, C4.5 and genetic programming.

Kumano *et al*. [54] investigated real-time application identification in encrypted traffic. IPsec and PPTP encryption were applied to web, interactive and bulk transfer flows to create an evaluation dataset. C4.5 and SVM algorithms were utilized to classify the application on these datasets. The authors then tested the accuracy of the classification using a different number of packets from the start

DOI: 10.1002/nem

of the flows. They measured the impact of using fewer packets on the flow features and proposed using features that show the least change to classify applications at an early stage.

6.2. Semi-supervised machine learning methods

Bernaille and Teixeira [55] used traffic clustering to detect applications encrypted by SSL. Their method has three steps. First, they detected SSL connections using a clustering algorithm (Gaussian mixture model) on packet sizes and directions of initial packets of a connection. The first three packets and 35 clusters provide good accuracy in detecting SSL traffic. After the SSL traffic is identified, the first data packets of the connections are identified. The sizes of the data packets are used by a clustering algorithm to detect an underlying application in the third step. However, the packet sizes are modified in the last step to allow for encryption overhead. The evaluation of the proposed method is carried out on traffic traces from live networks and a manually generated packet trace. The datasets contain HTTP, POP3, FTP, BitTorrent and eDonkey application protocols encrypted using SSL. The authors also show that using a combination of clustering and port numbers to differentiate between applications in clusters provides better results than clustering alone.

Maolini *et al*. [56] identify SSH traffic and determine underlying protocols (SCP, SFTP and HTTP) using a k-means algorithm. Only three packet features are used: the direction, the number of bytes and the timestamps of each packet. Their private datasets are created from artificial traffic and contain HTTP, FTP, POP3 and SSH protocols. Control packets such as TCP handshake, retransmitted packets and ACK-only packets are removed from statistics as they negatively affect the precision. Authors use only first three to seven packets to achieve a real-time identification.

Backquet *et al*. [57,58] use a MOGA to select a flow feature subspace and parameters for a clustering algorithm, which detects encrypted traffic. The second work employs a hierarchical k-means algorithm to increase the identification's accuracy. The authors evaluate both approaches on a private dataset captured at a university campus. SSH is used as a representative of encrypted traffic, and the ground truth is gained from the payload of the captured packets. Based on previous works, the authors argue that the feature selection and number of clusters highly affect the overall accuracy. Therefore, the authors selected four objectives for the genetic algorithm: cluster cohesiveness, cluster separation, the number of clusters and the amount of used flow features. The results show (a) that only 14 from a total of 38 flow features were used by the best-performing algorithm and (b) that using a hierarchical k-means algorithm increases the identification performance.

Bar-Yanai *et al*. [59] combined k-means and k-nearest neighbour clustering algorithms to construct a new, real-time classifier for encrypted traffic. The resulting classification algorithm has the light weight complexity of the k-means algorithm and accuracy of the k-nearest neighbour algorithm. They claim the method is fast, accurate and robust in regard to encryption, asymmetric routing and packet ordering. A labelled dataset was prepared from generated samples of the traffic, and the classification of the dataset was carried out using payloads of the packets. Flows shorter than 15 packets were removed from the dataset, as real-time classification for such short flows is not of practical use. If available, the first 100 packets were used for the classification. The authors stress that their method is applicable in a real-time environment and tested their implementation on an ISP link. The application protocols classified are HTTP, SMTP, POP3, Skype, eDonkey, BitTorrent, Encrypted BitTorrent, Real-time Transport Protocol and ICQ.

Zhang *et al*. [60] propose an improvement to the k-means clustering algorithm. Using harmonic mean to reduce the impact of random initial clustering centres, the authors are able to increase the accuracy of the k-means clustering algorithm for classifying encrypted traffic. The authors test their approach on two datasets. The first contains only data labelled SSH and non-SSH, and the second contains traffic from Skype, QQ, SSH, SSL and MSN protocols. However, the selection of the datasets and the selection of flow features used for classification were not justified in the paper.

Du and Zhang [61] used a k-means algorithm to discern traffic of three BitTorrent clients. Flow and packet header features, such as IP addresses, ports, numbers and the length of packets, were taken into consideration. The authors generated the traffic manually, and, therefore, their data sample contains only traffic from three clients. The authors highlight that the method is fast and simple and that it can be used to identify the traffic in real time.

DOI: 10.1002/nem

6.3. Basic statistical methods

De Montigny-Leboeuf [62] described the process of identifying traffic using flows and flow features. The author showed how to derive the flow features he uses and how to use them to identify the application type (e.g. interactive typing and data transfer) and block ciphers. Moreover, the author provides a list of recognition criteria for HTTP and HTTPS web browsing traffic, IMAP, POP, SMTP, SSH, Telnet, rlogin, FTP command and data, MSN chat and TCP audio streams. A subset of 39 traffic features was used to identify each application.

Wang *et al*. [63] computed entropy from packet payloads and used it for classifying traffic. They differentiate between eight different traffic classes: text, picture, video, audio, Base64 encoded text, Base64 encoded image, compressed and encrypted. The entropy is computed on chunks of different lengths, and the authors used an SVM algorithm for selecting feature. The computation of the entropy for four different chunk lengths was found to be sufficient for accuracy and performance. The most difficult task was to separate encrypted and compressed traffic. The authors provided an additional heuristic to distinguish these categories using frequencies of four-bit characters. The datasets used are manually obtained from captured network communication and processed to contain traffic from all classes.

Korczynski and Duda [64] designed a method called Statistical Protocol Identification to identify types of communication (voice call, SkypeOut, video conference, chat, file upload and file download) in Skype traffic. Nine flow features were selected using forward selection, which evaluates how a given feature improves the classification performance. A private, artificially created dataset with Skype traffic was used. Other traffic with SSL, SSH, HTTP, SCP, SFTP, VoIP, BitTorrent and other services was used to test the robustness of the method. The authors reported high accuracy in their method, although the distinction between voice and video traffic remains a difficult problem.

Amoli and Hamalainen [65] described a Network Intrusion Detection System (NIDS) capable of detecting attacks in encrypted network traffic in real time. The NIDS has two engines; the first one uses network change measurement to detect changes in a time series, such as denial of service and distributed denial of service. The first engine clusters the data to lessen the load on the second engine. The goal of the second engine is to detect the bot master behind the attack. The second engine clusters the prepared data to obtain the behaviour of the attackers and compares it with historical data. The authors used the detected anomalies to identify the botnet masters.

Korczynski and Duda [66] proposed using stochastic fingerprints based on Markov chains for identifying application traffic in SSL/TLS sessions. The method is payload based and uses statistical information from SSL/TLS headers. The authors tested their method on 12 representative applications such as Twitter, Skype and Dropbox. The Markov chain fingerprints are based on protocol specific distributions of packets in time. Datasets were captured from real traffic, contain only SSL/TLS traffic and were not published. The ground truth was obtained by inspecting domain names of the SSL/TLS traffic. The authors discovered that many protocol implementations differ from the RFC specification, which required them to adjust the fingerprints.

6.4. Hybrid methods

Karagiannis *et al*. [67] focused on host-based classification. Their method uses only information from the network level and therefore is not affected by transport layer encryption (e.g. TLS). The authors classified the behaviour of the hosts on social, functional and application levels, without access to packet payloads or headers. Each level was classified independently, and a cross-level classification was performed afterwards. Particular applications were represented using graphlets (Figure 7), which are representations of the application's behaviour. The authors then used heuristics to refine the classification. The ground truth for captured datasets was established by using a signature-based payload classification. Without using transport layer information, only the following traffic classes were identified: web, p2p, data (FTP, database), network management (DNS, SNMP, NTP), mail (SMTP, POP, IMAP), news (NNTP), chat (IRC, AIM, MSN messenger), streaming and gaming.

Wright *et al*. [68] worked on a classification of traffic in encrypted tunnels. Multiple flows can be wrapped in a single flow representing the encrypted tunnel. The information from the packet headers was not applicable; therefore, the authors used only packet sizes, timing and communication

DOI: 10.1002/nem

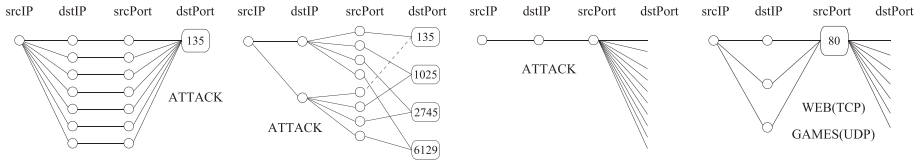

Figure 7. A visual representation of transport-layer interactions for various applications [67]

direction. A k-nearest neighbour classifier was used for classification when all TCP connections in a set carried the same application protocol. When TCP connections carried different application protocols, the authors used hidden Markov models. The authors also demonstrated that it is possible to determine the number of flows in an encrypted tunnel. The port numbers were used to obtain a ground truth for the captured dataset. The authors argue that mislabelled data only decreased the efficiency of their classification algorithm, and therefore, the real accuracy would be even higher than the reported one. The classifiers were able to detect the following application protocols: HTTP, HTTPS, SMTP, AIM, FTP, SSH and Telnet.

Koch and Rodosek [35] proposed a system for detecting interactive attacks using SSH. Packet sizes, IP addresses and packet inter-arrival times were used to create clusters of packets, which were likely to match an SSH command and its corresponding response. The SSH protocol was recognized based on the port number, and individual commands were identified from the clusters. Following this, sequences of commands were evaluated, and possible malicious sequences were reported. The system allows for the customization of malicious sequences' definitions using a sub-goals characterization. Each sub-goal maps to a malicious event, such as data gathering or system manipulation. The results from the evaluation of the proposed method show that such identification is possible.

Khakpour and Liu [69] used an entropy of packet payloads for classifying traffic. The authors showed how to compute the entropy of files and how to modify the formula for on-the-fly computation. Several entropy values were computed for each packet. Classification And Regression Tree and SVM methods were used for a subsequent classification based on the computed values. Their results demonstrated that SVM methods provide comparable accuracy with less false positives. The authors argue that it is necessary to exclude application layer headers such as HTTP response or picture headers. The reason for this is that computing entropy on the headers leads to a bias and misclassification of a packet. Therefore, a cut-off threshold was used to strip application headers from unknown protocols. The traffic was first classified into three categories: text, encrypted or binary. The authors also postulated that the classification can be more fine grained, and they investigated the classification of application protocols. Then, they demonstrated that it is possible to determine an encryption algorithm with a higher accuracy than random guessing, which they found surprising.

6.5. A summary of machine learning and statistical encrypted traffic classification

We have provided a summarizing overview of the feature-based traffic classification papers and methods they use in Table 4. Where a method belongs to multiple categories, it is not marked as a hybrid, but all the categories are listed instead. We find this approach more descriptive than using a hybrid category as defined by the taxonomy.

Most surveyed methods use flow or packet header features as an input for the classification techniques. The authors of [50,53] compare the results gained by utilizing packet header features and flow features. They show that using both sets of features can result in faster and more accurate classification algorithms. Nevertheless, using all the available traffic features does not necessarily lead to the best classification performance as demonstrated by the authors of [48].

The column *Number of features* shows how many flow or packet header features were used in each method. Some methods used different subsets of the features for different algorithms, and this is denoted by the $\subseteq$ mark. Moreover, some of the methods used a feature selection algorithm to select the best combination of features from the entire feature set. These methods are marked in the *Feature selection* column. For this case, the *Number of features* column represents the initial number of features.

DOI: 10.1002/nem

Table 4. A summary table of cited papers and methods they use to detect encrypted traffic

Reference	Publication year	Input					Number of features	Feature selection	Technique								Output	Encrypted protocol ident.	Real-time identification	Data set				Ground truth
		Traf. Payload	Traf. Properties						Payload Ins.	Graphlets	Statis.		Machine			Method(s)				Public	Private	Real	Artificial	
			Host Com.	Host	Flow	Packet					Basic	Heuristic	Sup.	Non Sup.	Semi Sup.									
[26]	2005				✓	✓	$\subseteq 39$				✓						F → AP, AT				✓	✓		port
[44]	2005		✓	✓	✓		–			✓	✓	✓					H → AT	✓			✓	✓		signature
[69]	2006				✓	✓	3						✓		✓	HMM, k-nearest neighbor	F → AP	✓			✓	✓		port
[16]	2007					✓	3								✓	Gaussian Mixture Model	P → AP	✓	✓		✓	✓		signature, known
[5]	2007				✓		22						✓			AdaBoost, RIPPER	F → AT	✓		✓	✓	–		port, known
[53]	2009					✓	3								✓	k-means	F → AP	✓	✓		✓		✓	known
[7]	2009				✓		22						✓			AdaBoost, RIPPER, SVM, Naive Bayes, C4.5	F → AP			✓	✓	✓		port, signature
[6]	2009				✓	✓	$\subseteq 30$						✓			RIPPER, C4.5	F → AT				✓	✓	✓	signature, known
[4]	2009					✓	39	✓					✓			AdaBoost, C4.5, Genetic Programming	F → AP				✓	✓		signature, port
[13]	2009				✓		38	✓							✓	k-means, Genetic Programming	F → AP				✓	✓		signature
[8]	2010				✓	✓	22						✓			AdaBoost, C4.5, Genetic Programming	F → AP	✓			✓	✓	✓	signature, known
[15]	2010				✓		17								✓	k-nearest neighbor, k-means	F → AP		✓		✓	✓		signature, known
[67]	2010	✓			✓		6		✓				✓			Naive Bayes	F → AP	✓		✓	✓	✓		known
[49]	2010					✓	–				✓	✓					F → FG		✓	–				–
[14]	2011				✓		38	✓							✓	(hierarchical) k-means	F → AP				✓	✓		signature
[68]	2011	✓					7	✓			✓					Entropy	F → AT	✓	✓		✓		✓	known
[59]	2011				✓		49	✓					✓			Naive Bayes	F → AP	✓			✓		✓	known
[11]	2011				✓		$\subseteq 46$	✓					✓			C4.5, k-means, MOGA	F → AP			✓	✓	✓		signature, port
[9]	2011				✓	✓	61	✓					✓			AdaBoost, C4.5, Genetic Programming	F → TP			✓	✓	✓		signature, port
[50]	2012				✓		9	✓			✓					Statistical Protocol Identification	F → AP	✓			✓		✓	known
[72]	2013				✓		22								✓	k-means	F → AP			✓	✓	✓		signature
[29]	2013				✓	✓	7								✓	k-means	F → AS				✓		✓	known
[47]	2013	✓					10	✓			✓		✓			Entropy, SVM, CART	F → AT, AP	✓			✓	✓	✓	known
[10]	2013				✓	✓	23				✓			✓		Change point detection, Clustering	F → A			–				–
[52]	2014				✓		29	✓					✓			C4.5, SVM	F → AT	✓	✓		✓		✓	known
[51]	2014	✓				✓	–				✓					Markov chains	F → AP	✓			✓	✓		signature

Output column legend: A Anomaly; AP Application Protocol; AS Application Software; T Application Type; H Host; F Flow; FG Fine-Grained; TP Traffic Payload

DOI: 10.1002/nem

Most classification algorithms are based on machine learning. The category of supervised machine learning algorithms is represented by hidden Markov models, RIPPER, AdaBoost, SVMs, C4.5 and naive Bayes. Several works [49,51–53] compare these algorithms to establish which is the best for the task of classifying traffic. The C4.5 algorithm performs the best in several cases; however, genetic programming is reported to achieve the best results in [53]. The second most common algorithm category is the semi-supervised machine learning, which is dominated by clustering algorithms. The k-means algorithm is the most frequent in this category, and the k-nearest neighbour comes second. The popularity of k-means is due to its variability, which allows it to be fine tuned for various purposes. It is often combined with genetic algorithms to find the best setting. The authors of [63,69] use the entropy of packet payloads to classify traffic. Using simple statistical properties of the traffic is the third most common classification method. Other methods are rarely used, mainly because they cannot learn from labelled traffic and therefore require too much effort to set up.

The SSH protocol is heavily used as a classification example. The authors of [47–51,53,57,58,60] test their methods for recognizing SSH and non-SSH traffic. Maiolini *et al*. [56] take the classification one step further and identify the type of traffic encapsulated in an SSH connection. The authors of [45,55,66] use SSL/TLS traffic and identify underlying application protocols. As the SSL/TLS protocol is more general and is used to encrypt various types of traffic, the complexity of identification is higher than for SSH. Another very popular protocol for identification is Skype, which is addressed by the authors of [51,53,64,66].

Some of the methods focus only on identifying encrypted traffic, whereas others try to identify the underlying application protocol. The methods that perform a more thorough analysis to gain information about the application protocol are indicated in the column *Encrypted protocol identification.*

Because all methods, with the exception of [67], classify whole flows and rely mostly on flow features, they are rarely able to classify traffic in real time. However, the authors of [54–56,59,63] achieved near real-time classification by extracting features of only a fixed number of packets in a flow. They argue that the first packets carry enough information for classification. Using a higher number of packets increases accuracy; therefore, it is possible to strike a balance between accuracy and early identification.

The *Dataset* columns describe whether the data used to evaluate the presented methods were taken from a live network (Real) or generated by a tool (Artificial). We also identify if the datasets were publicly available (Public), made available by the authors (Published) or kept undisclosed as they contain sensitive information (Private). If more than one dataset was used for each evaluation, we simply performed a union of the dataset descriptions. The *Ground truth* column indicates how the ground truth was obtained for each dataset. Common methods are based on port numbers or signatures, which use the packet payload. When the datasets are generated manually, the ground truth is known in advance.

The classification accuracy reported by authors of the surveyed methods depends heavily on the datasets used. All authors use their own private datasets, which are seldom published. Such methods simply cannot be compared without repeating the experiments on a common dataset. The authors of [45,47,49,51,53,60] also used publicly available datasets that were either labelled beforehand, using payload when available, or simply labelled using port numbers. A combination of datasets is often used to test the robustness of the methods.

The surveyed methods show that much effort was put into classifying encrypted traffic. We believe that there are several points that should be taken into account in any future research in this field. First, identifying encrypted traffic is not enough. The identification of the underlying protocol is the real challenge. Second, an SSL/TLS protocol should be used as the reference protocol, as it can contain much more complex traffic than the SSH protocol. Finally, the traces used should be labelled and made available to other researchers. Following these points does not limit the scope of future research; however, it simplifies the comparison of the presented approaches and allows others to verify the results more easily.

7. CONCLUSIONS

In this paper, we presented an overview of current approaches for the classification and analysis of encrypted traffic. First, we selected a number of the most widely used encryption protocols and

DOI: 10.1002/nem

described their packet structure and standard behaviour in a network. Second, we focused on information that is provided by encryption protocols themselves. We found that the initiation phase often provides information about the protocol version, ciphers used and the identity of at least one communicating party. Such information can be used to monitor and enforce security policies in an organization. We also discovered that the use of information from the unencrypted parts of an encrypted connection for a network anomaly detection is only briefly investigated by researchers. Information about communicating parties can be leveraged to discern the type of encrypted traffic. For example, the list of supported cipher suites provided by a client when establishing a secure connection can help to identify the client. We believe that the use of unencrypted parts from the initiation of an encrypted connection should be explored in more detail.

Before starting the analysis of the encrypted network traffic, it is necessary to identify it. Thus, we surveyed approaches to classifying network traffic. These were, first, payload-based methods, which use knowledge of a packets' structure, and feature-based methods, which use characteristics specific to the protocol flow. For the payload-based classification, there are several open-source traffic classifiers, which can identify encrypted traffic using pattern matching. The initiation of a communication often has a strictly defined structure; therefore, the patterns can be constructed for specific protocols. The main difference between various classifiers is that some of them require traffic from both directions of the communication to correctly classify the flows.

Feature-based traffic classifiers have been intensively researched over the last decade. Many statistical and machine-based learning methods have been applied to the task of traffic classification. Despite this, there are no conclusive results to show which method has the best properties. The main reason is that the results depend heavily on the datasets used and the configuration of the methods. We have applied the multilevel taxonomy of Khalife *et al*. [8] and categorized existing methods. Our results show that most of the authors use private datasets, sometimes in combination with public ones. For this reason, the individual results are not directly comparable. Most of the methods use supervised or semi-supervised machine learning algorithms to classify flows and even determine the application protocol of a given flow. Most methods target encryption protocols, such as SSH, SSL/TLS and encrypted BitTorrent, and use similar methods. However, there are also some novel works that apply innovative approaches to refine the classification up to deriving the content of the encrypted connections.

Most authors of feature-based classification methods claim that their approach is privacy sensitive as it does not require the traffic payload. However, privacy issues are much wider. In 2013, the Cybersecurity Research Ethics Dialog & Strategy Workshop [70] started a discussion about the influence of cyber-security research on the privacy of Internet users. Researchers need to keep in mind that their research activities have a significant impact on infrastructure security, network neutrality and privacy of end-users.

In the past, Internet protocols were not designed with security considerations in mind. The recent interest in privacy has motivated the Internet Engineering Task Force (IETF) to reconsider this approach and discuss the privacy aspects of the protocols. Discussions held in [71] revealed that monitoring privacy issues are of great concern. This discussion resulted in a new RFC [72], where the IETF clearly states that pervasive monitoring is considered to be an attack. The document suggests that the IETF's protocols should be hardened against such monitoring. It is clear that the struggle between the demand for privacy and the need for security is still beginning.

ACKNOWLEDGEMENT

This material is based on the work supported by the Security Research for the Needs of the State 2010–2015 programme funded by the Ministry of the Interior of the Czech Republic.

REFERENCES

1. Sandvine, Inc. *Global Internet Phenomena Report 1H 2014*, 2014. Available from: https://www.sandvine.com/downloads/general/global-internet-phenomena/2014/1h-2014-global-internet-phenomena-report.pdf [30 November 2014].
2. Dainotti A, Pescape A, Claffy KC. Issues and future directions in traffic classification, *Network, IEEE* 2012; **26**(1): 35–40.
3. Nguyen TTT, Armitage G. A survey of techniques for internet traffic classification using machine learning, *Communications Surveys & Tutorials, IEEE* 2008; **10**(4): 56–76.

DOI: 10.1002/nem

4. Zhang M, John W, Claffy KC, Brownlee N. State of the art in traffic classification: a research review. In *Pam '09: 10th International Conference on Passive and Active Measurement, Student Workshop,* Seoul, Korea, 2009.
5. Callado A, Kamienski C, Szabo G, Gero B, Kelner J, Fernandes S, Sadok D. A survey on internet traffic identification. *Communications Surveys & Tutorials, IEEE* 2009; **11**(3): 37–52.
6. Finsterbusch M, Richter C, Rocha E, Muller J-A, Hanssgen K. A survey of payload-based traffic classification approaches. *Communications Surveys & Tutorials, IEEE* 2014; **16**(2): 1135–1156.
7. Cao Z, Xiong G, Zhao Y, Li Z, Guo L. A survey on encrypted traffic classification. In *Applications and Techniques in Information Security*, Batten L, Li G, Niu W, Warren M (eds.), Communications in Computer and Information Science, vol. 490, Springer Berlin Heidelberg: Berlin, Germany, 2014; 73–81.
8. Khalife J, Hajjar A, Diaz-Verdejo J. A multilevel taxonomy and requirements for an optimal traffic-classification model. *International Journal of Network Management* 2014; **24**(2): 101–120.
9. ISO. *ISO/IEC 7498-1:1994 Information technology—Open Systems Interconnection—Basic Reference Model: The Basic Model*, 2nd edn., International Organization for Standardization: Geneva, Switzerland, 1994.
10. Frankel SE, Kent K, Lewkowski R, Orebaugh AD, Ritchey RW, Sharma SR. Guide to IPsec VPNs. *Technical Report SP 800-77*, National Institute of Standards & Technology: Gaithersburg, MD, United States, 2005.
11. Kaufman C, Hoffman P, Nir Y, Eronen P. Internet Key Exchange Protocol Version 2 (IKEv2). *Technical Report RFC 5996 (Proposed Standard)*, Internet Engineering Task Force, 2010. Updated by RFCs 5998, 6989.
12. Kent S. IP Encapsulating Security Payload (ESP). *Technical Report RFC 4303 (Proposed Standard)*, Internet Engineering Task Force, 2005.
13. Dierks T, Rescorla E. *The Transport Layer Security (TLS) Protocol Version 1.2*, IETF: Internet Engineering Task Force, 2008. Updated by RFCs 5746, 5878, 6176.
14. Freier A, Karlton P, Kocher P. The Secure Sockets Layer (SSL) Protocol Version 3.0. *Technical Report RFC 6101 (Historic)*, Request for Comments, Internet Engineering Task Force, 2011.
15. Meyer C. 20 Years of SSL/TLS Research. An Analysis of the Internet's Security Foundation. *Ph.D. Thesis*, 2014.
16. Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. *Technical Report RFC 5280 (Proposed Standard)*, Internet Engineering Task Force, 2008. Updated by RFC 6818.
17. Ylonen T, Lonvick C. The Secure Shell (SSH) Transport Layer Protocol. *Technical Report RFC 4253 (Proposed Standard)*, Internet Engineering Task Force, 2006. Updated by RFC 6668.
18. Galbraith J, Saarenmaa O. *SSH File Transfer Protocol draft-ietf-secsh-filexfer-13.txt. Internet-Draft*, 2006.
19. Pechanec J. *How the SCP protocol works. Jan Pechanec's Weblog. Weblog post*, 2007. Available from: https://blogs.oracle.com/janp/entry/how_the_scp_protocol_works [30 November 2014].
20. Stallings W. Protocol basics: Secure Shell protocol. *The Internet Protocol Journal* 2009; **12**(4): 18–30.
21. Harrison D. *Index of BitTorrent Enhancement Proposals. Web page*, 2014. Available from: http://www.bittorrent.org/beps/bep_0000.html [30 November 2014].
22. Azureus Software Inc. *Message Stream Encryption. Vuze Wiki. Web page*, 2014. Available from: http://wiki.vuze.com/w/Message_Stream_Encryption [30 November 2014].
23. Skype and Microsoft. *Skype. Web page*, 2014. Available from: http://www.skype.com/ [30 November 2014].
24. Adami D, Callegar C, Giordano S, Pagano M, Pepe T. Skype-Hunter: A real-time system for the detection and classification of Skype traffic. *International Journal of Communication Systems* 2011; **25**(3): 386–403.
25. Skype Limited. *Skype Connect™ Requirements Guide. Online*, 2011. Available from: http://download.skype.com/share/business/guides/skype-connect-requirements-guide.pdf [30 November 2014].
26. IANA—Internet Assigned Numbers Authority. *Protocol Registries. Web page*, 2014. Available from: http://www.iana.org/protocols [30 November 2014].
27. Qualys, Inc. *HTTP Client Fingerprinting Using SSL Handshake Analysis. Web page*, 2014. Available from: https://www.ssllabs.com/projects/client-fingerprinting/ [30 November 2014].
28. Majkowski M. *SSL fingerprinting for p0f. Web page*, 2012. Available from: https://idea.popcount.org/2012-06-17-ssl-fingerprinting-for-p0f/ [30 November 2014].
29. Holz R, Braun L, Kammenhuber N, Carle G. The SSL landscape: a thorough analysis of the X.509 PKI using active and passive measurements. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11. ACM: New York, NY, USA, 2011; 427–444.
30. Durumeric Z, Kasten J, Bailey M, Halderman JA. Analysis of the HTTPS certificate ecosystem. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13. ACM: New York, NY, USA, 2013; 291–304.
31. sslbl.abuse.ch. *SSL Blacklist.* Web page, 2014. Available from: https://sslbl.abuse.ch/ [30 November 2014].
32. Blake-Wilson S, Nystrom M, Hopwood D, Mikkelsen J, Wright T. Transport Layer Security (TLS) Extensions. *Technical Report RFC 4366 (Proposed Standard)*, Internet Engineering Task Force, 2006. Obsoleted by RFCs 5246, 6066, updated by RFC 5746.
33. Huang L, Adhikarla S, Boneh D, Jackson C. An experimental study of TLS forward secrecy deployments. *Internet Computing, IEEE* 2014; **18**(6): 43–51.
34. Miller B, Huang L, Joseph AD, Tygar JD. I know why you went to the clinic: risks and realization of HTTPS traffic analysis. In *Privacy Enhancing Technologies*, Lecture Notes in Computer Science, vol. 8555, Springer International Publishing: New York, USA, 2014; 143–163.
35. Koch R, Rodosek GD. Command evaluation in encrypted remote sessions. In *Proceedings of the 2010 Fourth International Conference on Network and System Security*, NSS '10. IEEE Computer Society: Washington, DC, USA, 2010; 299–305.

DOI: 10.1002/nem

36. Hellemons L, Hendriks L, Hofstede R, Sperotto A, Sadre R, Pras A. SSHCure: a flow-based SSH intrusion detection system. In *Dependable Networks and Services*, Lecture Notes in Computer Science, vol. 7279, Springer Berlin Heidelberg: Berlin, Germany, 2012; 86–97.
37. ipoque GmbH. *PACE 2.0. Web page*. Available from: http://www.ipoque.com/en/products/pace [30 November 2014].
38. Cisco Systems, Inc. *Network Based Application Recognition (NBAR). Web page*, 2014. Available from: http://www.cisco.com/c/en/us/products/ios-nx-os-software/network-based-application-recognition-nbar [30 November 2014].
39. Deri L, Martinelli M, Bujlow T, Cardigliano A. nDPI: Open-source high-speed deep packet inspection, In *Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International,* Nicosia, Cyprus, 2014; 617–622.
40. Alcock S, Nelson R. *Libprotoident: traffic classification using lightweight packet inspection. Online article*, 2012. Available from: http://www.wand.net.nz/~salcock/lpi/lpi.pdf [30 November 2014].
41. ClearFoundation. *l7-filter. Web page*, 2013. Available from: http://l7-filter.clearfoundation.com/ [30 November 2014].
42. Bujlow T, Carela-Español V, Barlet-Ros P. Independent comparison of popular DPI tools for traffic classification. *Computer Networks* 2015; **76**(0): 75–89.
43. Moore A, Crogan M, Zuev D. Discriminators for use in flow-based classification, Queen Mary, University of London, 2005.
44. Alpaydin E. *Introduction to Machine Learning*, 2nd edn., The MIT Press: London, England, 2010.
45. Sun G, Xue Y, Dong Y, Wang D, Li C. An novel hybrid method for effectively classifying encrypted traffic, In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE,* Miami, Florida, USA, 2010; 1–5.
46. Okada Y, Ata S, Nakamura N, Nakahira Y, Oka I. Application identification from encrypted traffic based on characteristic changes by encryption. In *2011 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR),* Naples, Florida, USA, 2011; 1–6.
47. Arndt DJ, Zincir-Heywood AN. A comparison of three machine learning techniques for encrypted network traffic analysis. In *2011 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA),* Paris, France, 2011; 107–114.
48. Alshammari R, Lichodzijewski PI, Heywood M, Zincir-Heywood AN. Classifying SSH encrypted traffic with minimum packet header features using genetic programming. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, GECCO '09. ACM: New York, NY, USA, 2009; 2539–2546.
49. Alshammari R, Zincir-Heywood AN. A flow based approach for SSH traffic detection. In *IEEE International Conference on Systems Man and Cybernetics, 2007. ISIC,* Montreal, Quebec, Canada, 2007; 296–301.
50. Alshammari R, Zincir-Heywood AN. A preliminary performance comparison of two feature sets for encrypted traffic classification, In *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems CISIS'08*, Advances in Soft Computing, vol. 53, Springer Berlin Heidelberg: Genoa, Italy, 2009; 203–210.
51. Alshammari R, Zincir-Heywood AN. Machine learning based encrypted traffic classification: identifying SSH and Skype. In *IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009,* Ottawa, Canada, 2009; 1–8.
52. Alshammari R, Zincir-Heywood AN. An investigation on the identification of VoIP traffic: case study on Gtalk and Skype. In *2010 International Conference on Network and Service Management (CNSM),* Niagara Falls, Canada, 2010; 310–313.
53. Alshammari R, Zincir-Heywood AN. Can encrypted traffic be identified without port numbers, IP addresses and payload inspection? *Computer Networks* 2011; **55**(6): 1326 –1350.
54. Kumano Y, Ata S, Nakamura N, Nakahira Y, Oka I. Towards real-time processing for application identification of encrypted traffic. In *2014 International Conference on Computing, Networking and Communications (ICNC),* Honolulu, Hawaii, USA, 2014; 136–140.
55. Bernaille L, Teixeira R. Early recognition of encrypted applications. In *Proceedings of the 8th International Conference on Passive and Active Network Measurement*, PAM'07. Springer-Verlag: Berlin, Heidelberg, 2007; 165–175.
56. Maiolini G, Baiocchi A, Iacovazzi A, Rizzi A. Real time identification of SSH encrypted application flows by using cluster analysis techniques. In *Networking 2009*, Fratta L, Schulzrinne H, Takahashi Y, Spaniol O (eds.), Lecture Notes in Computer Science, vol. 5550, Springer Berlin Heidelberg: Berlin, Germany, 2009; 182–194.
57. Bacquet C, Zincir-Heywood AN, Heywood MI. An investigation of multi-objective genetic algorithms for encrypted traffic identification. In *Computational Intelligence in Security for Information Systems*, Advances in Intelligent and Soft Computing, vol. 63, Springer Berlin Heidelberg, 2009; 93–100.
58. Bacquet C, Zincir-Heywood AN, Heywood MI. Genetic optimization and hierarchical clustering applied to encrypted traffic identification. In *2011 IEEE Symposium on Computational Intelligence in Cyber Security (CICS),* Paris, France, 2011; 194–201.
59. Bar-Yanai R, Langberg M, Peleg D, Roditty L. Realtime classification for encrypted traffic. In *Experimental Algorithms*, Lecture Notes in Computer Science, vol. 6049, Springer Berlin Heidelberg: Berlin, Germany, 2010. 373–385.
60. Zhang M, Zhang H, Zhang B, Lu G. Encrypted traffic classification based on an improved clustering algorithm. In *Trustworthy Computing and Services*, Communications in Computer and Information Science, vol. 320, Springer Berlin Heidelberg: Berlin, Germany, 2013; 124–131.
61. Du Y, Zhang R. Design of a method for encrypted P2P traffic identification using k-means algorithm. *Telecommunication Systems* 2013; **53**(1): 163–168.
62. De Montigny-Leboeuf A. Flow attributes for use in traffic characterization, *Communications Research Centre Canada, Tech. Rep*, 2005.
63. Wang Y, Zhang Z, Guo L, Li S. Using entropy to classify traffic more deeply. In *2011 6th IEEE International Conference on Networking, Architecture and Storage (NAS),* Dalian, Liaoning, China, 2011; 45–52.

DOI: 10.1002/nem

64. Korczynski M, Duda A. Classifying service flows in the encrypted Skype traffic. In *2012 IEEE International Conference on Communications (ICC),* Ottawa, Canada, 2012; 1064–1068.
65. Amoli PV, Hamalainen T. A real time unsupervised NIDS for detecting unknown and encrypted network attacks in high speed network. In *2013 IEEE International Workshop on Measurements and Networking Proceedings (M&N),* Naples, Italy, 2013; 149–154.
66. Korczynski M, Duda A. Markov chain fingerprinting to classify encrypted traffic. In *Infocom, 2014 Proceedings IEEE*, 2014; 781–789.
67. Karagiannis T, Papagiannaki K, Faloutsos M. BLINC: Multilevel Traffic Classification in the Dark, In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '05. ACM: New York, NY, USA, 2005; 229–240.
68. Wright CV, Monrose F, Masson GM. On inferring application protocol behaviors in encrypted network traffic. *Journal of Machine Learning Research* 2006; **7**: 2745–2769.
69. Khakpour AR, Liu AX. An information-theoretical approach to high-speed flow nature identification. *Networking, IEEE/ACM Transactions on* 2013; **21**(4): 1076–1089.
70. CAIDA. *Cyber-security Research Ethics Dialog & Strategy Workshop*, 2013. Available from: http://www.caida.org/workshops/creds/1305/ [30 November 2014].
71. IETF. *IETF 88 Proceedings, Technical Plenary*, 2014. Available from: http://www.ietf.org/proceedings/88/technical-plenary.html [30 November 2014].
72. Farrell S, Tschofenig Hb. Pervasive Monitoring Is an Attack. *RFC 7258 (Best Current Practice)*, IETF: Internet Engineering Task Force.

AUTHORS' BIOGRAPHIES

Petr Velan works at Masaryk University as a researcher at the Network Traffic Analysis Group of the Institute of Computer Science and is a PhD candidate at the Faculty of Informatics. He participated in several projects including Czech CyberCrime Centre of Excellence and Cybernetic Proving Ground. His main research interest is application visibility in network monitoring and its application to network security.

Milan Čermák is a security researcher at the Computer Security Incident Response Team of Masaryk University (CSIRT-MU) and a PhD candidate in Computer Systems and Technologies at the Faculty of Informatics, Masaryk University. He participated in several projects, including Cybernetic Proving Ground and Technology for processing and analysis of network data in big data concept. His main research interest is a network-based intrusion detection, a similarity search-based analysis of network data and a DNS traffic analysis.

Pavel Čeleda is an associate professor affiliated with the Institute of Computer Science at the Masaryk University in Brno, Czech Republic. He received a PhD degree in Informatics from University of Defence, Brno. His main research interests include flow monitoring, cyber security and design of network security devices.

Martin Drašar is the head of Network Analysis Group and a security researcher at the Institute of Computer Science of Masaryk University. His primary research interests are collaborative adaptive network attacks and distributed intrusion detection systems. He defended his PhD thesis 'Behavioral Detection of Distributed Dictionary Attacks' in April 2015.

DOI: 10.1002/nem