Security Monitoring of IoT Communication Using Flows

Petr Matoušek
matousp@fit.vutbr.cz
Brno University of Technology
Brno, Czech Republic

Ondřej Ryšavý
rysavy@vutbr.cz
Brno University of Technology
Brno, Czech Republic

Matěj Grégr
mgregr@fit.vutbr.cz
Brno University of Technology
Brno, Czech Republic

ABSTRACT

Network monitoring is an important part of network management that collects valuable metadata describing active communication protocols, network transmissions, bandwidth utilization, and the most communicating nodes. Traditional IP network monitoring techniques include the SNMP system, flow monitoring, or system logging. The environment of the Internet of Things (IoT) networks, however, shows that these approaches do not provide sufficient visibility of IoT communication which would allow network administrators to identify possible attacks on IoT nodes. The reason is obvious: IoT devices lack sufficient computational resources to fully implement monitoring agents, LAN IoT data communication is often directly over data link layers rather than IP, and IoT sensors produce an endless flow of small packets which can be difficult to process in real-time. To tackle these limitations we propose a new IoT monitoring model based on extended IPFIX records. The model employs a passive monitoring probe that observes IoT traffic and collects metadata from IoT protocols. Using extended IPFIX protocol, flow records with IoT metadata are sent to the collector where they are analyzed and used to provide a global view on the whole IoT network and its communication. We also present two statistical approaches that analyze IoT flows data in order to detect security incidents or malfunctioning of a device. The proof-of-concept implementation is demonstrated for Constrained Application Protocol (CoAP) traffic in the smart home environment.

CCS CONCEPTS

• **Networks** → **Network security**; • **Security and privacy** → *Intrusion/anomaly detection and malware mitigation*; • **Computer systems organization** → *Embedded and cyber-physical systems*;

KEYWORDS

Internet of Things, security, monitoring, statistical anomaly detection, IPFIX, CoAP

ACM Reference format:
Petr Matoušek, Ondřej Ryšavý, and Matěj Grégr. 2019. Security Monitoring of IoT Communication Using Flows. In *Proceedings of ECBS19: 6th Conference on the Engineering of Computer Based Systems, Bucharest, RO, Sep 02–03, 2019 (ECBS19),* 9 pages.
https://doi.org/https://doi.org/10.1145/3352700.3352718

ECBS19, Sep 02–03, 2019, Bucharest, RO

ACM ISBN 978-1-4503-7636-5…$15.00
https://doi.org/https://doi.org/10.1145/3352700.3352718

1 INTRODUCTION

Growing deployment of smart homes and buildings with tens or hundreds of small sensors producing an endless flow of monitoring data brings a new challenge for network monitoring. Traditional approaches of IP monitoring based on Simple Network Management Protocol (SNMP) [9], flow monitoring [10] or system logging [19] do not provide sufficient visibility of IoT traffic transmitted among IoT devices and the gateway, or between the gateway and the cloud, see Figure 1.

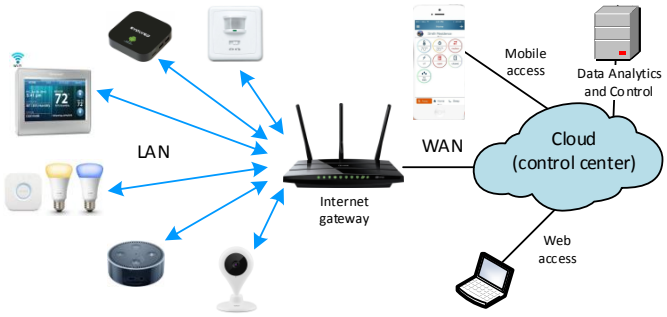

Figure 1: Topology of IoT ecosystem.

This limits ability of detecting attacks on the IoT network and unusual behavior of connected devices caused by malfunctioning or misconfiguration.

The main reason for inadequate monitoring of IoT networks is the fact that IoT end nodes (sensors, actuators, smart appliances) lack sufficient computational resources that are required to implement an SNMP agent or Syslog client. Moreover, IoT nodes often communicate directly over Layer 1 and 2 (physical and data link) technologies such as Z-Wave, ZigBee, 6LoWPAN, IEEE 802.15.4 (wireless PAN) or IEEE 802.11 WiFi [24] while traditional network monitoring is based on IP protocol and requires the full TCP/IP stack.

Another features of IoT communication is an ceaseless sequence of packets that transmit status and measured quantities of IoT nodes to the IoT control center that is mostly located in the cloud outside the local network. Sensors like a thermometer or PIR sensor generate tens of packets which causes high requirements on traffic monitoring and data processing.

Due to IoT node's hardware limitation IoT communication is mostly transmitted unencrypted and without authentication which makes it vulnerable to different kinds of attacks including IoT resource scanning, command injection, unauthorized access or DDoS attack [8, 31, 37]. In 2016, thousands of IP cameras were infected by

Mirai malware and became an involuntary source of DDoS attack against DNS provider Dyn [3, 8]. Flaws in IoT devices can be misused by attackers as documented on the case of Comcast's Xfinity Home Security system[1] where IoT sensors falsely reported that windows and doors are closed while they have been opened by the attacker.

To identify such security incidents on the IoT network, enhanced IoT monitoring providing IoT visibility is needed. This includes extracting meta data of IoT communication that describe types of transmitted IoT packets (monitoring data, control commands, keep-alive message), detect active IoT nodes and their resources, and observe typical size and frequency of sent packets and probability of their occurrence. Such meta data cannot be obtained using traditional network monitoring that mostly operates on Layer 3 and 4 (network and transport layers) but requires processing of Layer 7 IoT protocols.

Our proposed approach is based on extended IPFIX monitoring [12, 39] which offers flexible definition of a network flow. In our case, IPFIX records are extended by L7 meta data obtained by the probe that monitors IoT communication on L7. Besides L3 and L4 information (e.g., src/dst IP addresses, src/dst ports), the probe also collects IoT meta data (e.g., IoT protocol, packet type, identification of requested IoT object, etc.). Such data are then analyzed and visualized on the IPFIX collector.

Enriched IPFIX monitoring significantly increases visibility of IoT communication and enables detection of common security incidents such as IoT network scanning, detection of rogue devices, spoofed message injection, unauthorized access or DDoS attack on the IoT network. In combination with anomaly detection (AD) techniques it helps to determine if a given IoT traffic is normal (expected) or if it is an attempt of an intruder to attack the network. Moreover, L7 visibility of IoT communication enables early detection of failures and misconfiguration of IoT devices communicating in the network.

Great advantage of the proposed IoT monitoring model is that it fully complies with traditional IPFIX monitoring and can be easily integrated into current network management systems.

1.1 Structure of the text

The paper is structured as follows. Section 2 gives an overview of current approaches related to IoT monitoring, discusses security issues of IoT communication and statistical methods for anomaly detection. Section 3 presents the proposed IoT monitoring model based on extended IPFIX records and its deployment in LAN environment where IoT data are transmitted directly over L1/L2, and on the edge where TCP/IP is available. Section 4 deals with IoT data analysis, namely with statistical-based flow analysis and statistical anomaly detection. We show how current methods can be extended and applied on IoT flows. The section presents results of our proof-of-concept implementation of CoAP monitoring [35] and data analysis using communication profiles. Last section summarizes benefits and limitations of our approach and shows how IoT monitoring can be deployed.

[1]See https://www.wired.com/2016/01/xfinitys-security-system-flaws-open- homes-to-thieves/ [March 2019]

1.2 Contribution

One contribution of the paper is an IoT monitoring model which includes redefinition of flows for the IoT context, parsing of IoT communication and extension of IPFIX flows by IoT header data. Using IoT-enabled IPFIX monitoring we increase IoT visibility that can be used for detecting security incidents. The second contribution includes two techniques for processing and analyzing IoT flow records: one is based on querying IoT flow record database and visualization of active IoT communication to the network administrator, the second approach presents statistical anomaly detection that creates communication profiles of IoT resources based on IoT flow data. These profiles are able to detect common cyber attacks against IoT networks and identify unusual behavior of connected devices. Feasibility of the presented approach is demonstrated on CoAP traffic.

2 STATE-OF-THE-ART

Traditional network monitoring system usually employ SNMP protocol [9] that proactively polls monitoring objects and gathers monitoring data. Flow-based network monitoring observes active communication and extract monitoring data from passing packets using Netflow [10] or IPFIX [12] probes. Flow *meta data* in the form of IPFIX records are later stored at the IPFIX collector. The third common approach uses event logging. A network device locally saves system events into a log file that is transmitted and processed by the Syslog server [19].

The above mentioned techniques are well-established for IP-based networks and Internet services, e.g., e-mail monitoring, file transfer, HTTP communication, or VoIP. However, these techniques do not usually provide sufficient application visibility for IoT communication, e.g., CoAP [35] or MQTT [28] protocols.

Network monitoring tools like Scrutinizer[2] or PRTG monitor[3] support IoT management only on Layer 3 and 4 without real visibility of IoT communication. IoT visibility is incorporated in NetScout TruView[4] to a certain extent. Cisco Application Visibility and Control (AVC) solution[5] also supports IoT data processing using Network-Based Application Recognition (NBAR) engine and Flexible Netflow [33], however, security processing and cyber attack detection is left on third party applications. Our work focuses on both visibility of IoT communication and security analysis of IoT flows.

Benefits of flow monitoring for network attack detection were presented in [16, 27]. Netflow monitoring data can be used for DDoS detection [25], botnet detection [2, 21], intrusion detection [40], or for even for application-aware monitoring [7]. As we show in this study, flow monitoring can be also applied to IoT networks but requires redefinition of flow and analysis of IoT protocols' headers.

Maggi and Vosseler [31] present common attacks against IoT communication and discuss the impact on functionality of both home IoT and industrial IoT networks. They show amplification attack on CoAP, invalid character insertion into MQTT messages, scanning attack on CoAP and MQTT, etc. Such attacks cannot be

[2]See www.plixer.com/products/scrutinizer [May 2019].
[3]See www.paessler.com/prtg [May 2019].
[4]See www.netscout.com/product/truview [May 2019]
[5]See www.cisco.com/c/en/us/td/docs/ios/solutions_docs/avc/guide/avc-user-guide.html [May 2019].

detected using traditional monitoring on Layer 3 since they require increased visibility of active IoT communication on Layer 7 which is addressed by this paper.

Statistical based anomaly detection is one of the widely used techniques [1]. The basic idea of statistical methods is to detect significant deviations of observed behavior from the normal one and mark it as anomalous or intrusions. Alpha-stable first-order model and statistical hypothesis testing were used to detect anomalies in network traffic by [36]. Manikopoulos and Papaavssiliou [32] developed a method that uses statistical models and multivariate classifiers in order to detect anomalous network conditions. Eskin in [17] proposes a mixture model enabling to detect anomalies from noisy data. His mixture model considers two probability distributions representing normal and anomalous events. Network monitoring and anomaly detection systems designed specifically for IoT environment commonly exploits characteristics of the IoT protocols. Granjal et al. [22] invented an IDS framework for the detection and prevention of attacks in the context of CoAP environments. Goldenberg and Wood [20] studied communication in industrial IoT and proposed DFA-based intrusion detection system that by observing the regular communication creates a detailed traffic model which is very sensitive to anomalies. In our work, we combine statistical fingerprinting approach proposed by [14] with multivariate classification of [32].

3 IOT MONITORING MODEL

First, we introduce the proposed IoT monitoring model that is based on IoT flows. Network monitoring usually includes three basic parts: (i) obtaining monitoring data, (ii) data processing and storage, and (iii) data analysis and presentation. The proposed IoT monitoring model covers all three parts, namely (i) observing IoT communication using an IoT-enabled IPFIX probe that creates IoT flow records, (ii) transmitting and storing IoT flow records in the IPFIX collector, and (iii) analyzing IoT flow records using statistical flow analysis and statistical anomaly detection methods.

3.1 IoT Flow Monitoring

The proposed model composes of an IoT-enabled IPFIX probe that monitors IoT traffic, parses headers of IoT protocols and extracts meta data from the headers that are useful for IoT network monitoring. In case of IoT traffic being transmitted directly over Layer 2, the probe should monitor this layer and add Layer 3 items into IoT flow records, see probe A at Figure 2. More typical deployment is to place the probe between the IoT hub and the edge router where the probe operates over TCP/IP, see probe B in Figure 2.

The IPFIX probe observes passing IoT packets, extracts selected data from IoT headers and inserts them into IoT-enabled IPFIX records. For each IoT protocol we define a specific set of headers useful for monitoring. Figure 3 shows an example of an IPFIX record enriched by CoAP header field values.

Traditional IP flow monitoring creates a flow record for a unidirectional IP flow having the same key header values, namely source and destination IP address, source and destination port, IP protocol, ToS and the incoming interface [33]. For IoT monitoring we define an *IoT flow* as a *set of packets passing a probe in both directions and having the same IoT property*, e.g., an IoT resource identifier, IoT

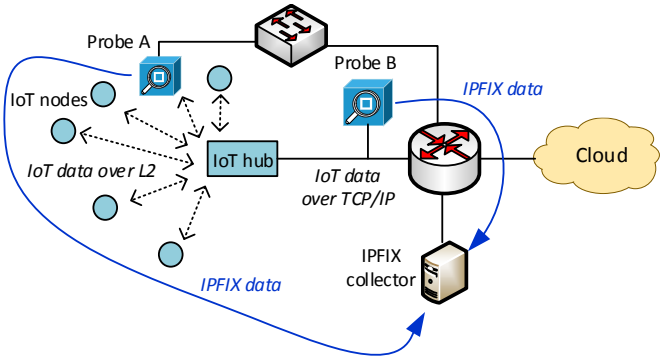

Figure 2: IoT monitoring using extended IPFIX.

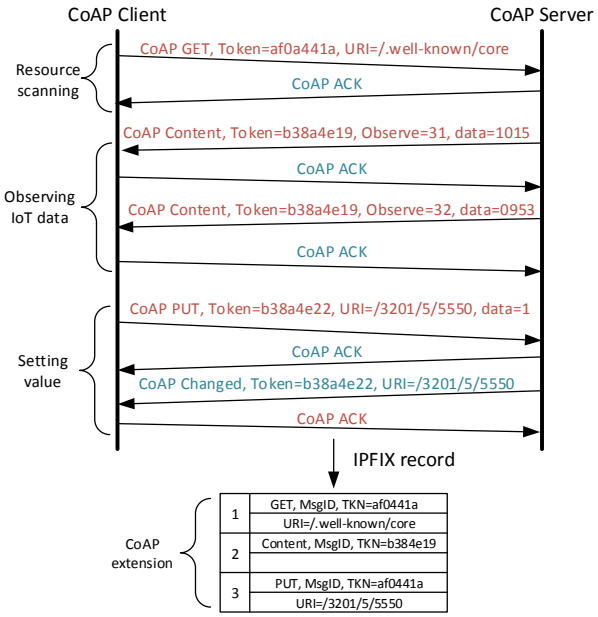

Figure 3: Extended IPFIX record for CoAP monitoring.

message type, etc. This enables us to aggregate sequences of same IoT packets that are exchange between two nodes. Packets with the same type and resource identifier form a single flow record which significantly reduces space for storing IoT monitoring data while providing sufficient visibility of IoT communication.

Figure 3 shows an example of three CoAP-enabled IPFIX records extended by CoAP message type (GET, Content, or PUT), message ID, token ID (TKN), and resource identifier in the form of URI. IoT packets having the same key properties create one IoT-extended IPFIX record that represents these packets.

Great advantage of the proposed monitoring model is that it can be easily integrated with the current IPFIX monitoring infrastructure. It requires only changes on the IPFIX monitoring probe which has to be extended by an protocol module that parses given IoT protocol, extracts packet headers and creates IoT flow records. IPFIX standard [12] supports flexible definition of IPFIX records which means that extracted IoT flow data can be incorporated into

a generic IPFIX flow record. Such extension is described using an IPFIX template that is sent from the probe to the collector together with IoT-extended IPFIX flow records. Thus, there is no need to make any changes on the collector. The collector interprets incoming IoT flow records using the template. The flow records are then stored at the collector for further analysis.

3.2 Using IoT Visibility for Security Monitoring

The main requirements for efficient management of IoT networks is visibility of IoT communication. IoT flow records provide such visibility. These records can be further used for detecting common security incidents as follows:

- *Resource scanning.* IoT protocols usually offer the possibility to list all available resources implemented on an IoT node (e.g., temperature, light, or humidity sensors) by one request. For example, if an MQTT client sends `"subscribe #"` request to the broker, the broker starts sending all MQTT messages received by the broker from all active nodes to that client. This legitimate feature, however, can be misused by an attacker to get a list of all active MQTT devices connected to the broker and all available resources. Similarly, a CoAP client can request available resources by sending `coap://address/.well-known/core` URI. With IoT monitoring, such requests can be detected and filtered out for specific stations.
- *DDoS attack.* Distributed Denial of Service (DDoS) attack means that an attacker frequently requests data from a legitimate IoT node in order to exhaust its resources by processing the received requests. When an attacker sends the requests with a spoofed IP address of existing IoT node, the answers to such requests may cause amplification attack on the target. For example, if an attacker sends CoAP PUT requests from a spoofed IP address and with the block-wise transfer mode set, an IoT node which IP address has been spoofed will be overloaded by big IoT packets with the answer. Such behavior can be detected by monitoring IoT commands.
- *Detection of a rogue device.* By long-term monitoring of IoT communication, a network administrator can create a list of legitimate IoT objects and resources. Based on the list, he can easily discover a new unauthorized IoT device that tries to communicate over the IoT network.
- *Unauthorized commands.* Most IoT protocols in LAN operate without authentication. By sending unauthorized commands, an attacker can manipulate with an IoT sensor, reconfigure IoT device, or switch off the devices or sensor. IoT monitoring makes transmitted IoT commands visible which helps detecting unauthorized commands.
- *Misusing IoT vulnerability.* Attackers often misuse software vulnerabilities of IoT devices for the attack. For example, by sending a disallowed UTF-8 encoded MQTT request, an attacker can disconnect the other MQTT clients from the broker[6]. By IoT monitoring, a network administrator can detect such behavior and reconfigure or update the device to prevent the attack.

[6]See vulnerability CVE-2017-7653 at cve.mitre.org [May 2019].

The above mentioned attacks on IoT communication show why IoT monitoring is important for network management. IoT visibility helps identifying active nodes, discovering attacks or detecting unauthorized behavior. Besides, historical IoT monitoring data are valuable for post-mortem analysis of past incidents.

4 IOT DATA ANALYSIS

Obtaining IoT flow records is the first phase of security monitoring. Next phase includes analysis of IoT flows with focus on visibility of IoT data transfers and detection of security threats [27]. In this section we present two approaches that work with IoT flows and give information about the IoT network: (i) *statistical-based flow analysis* that provides statistics of past IoT transfers, reveals active IoT resources on the network, and can be used as a source for building *whitelists* [34], and (ii) *statistical anomaly detection* that creates communication profiles for each of IoT resources during a learning phase and uses these profiles to determine if a current traffic corresponds to a known communication profile, i.e., it can be marked as normal, or if it significantly differs from known profiles and thus it will be marked as abnormal. We show how these techniques employ IoT flow records and can be used to detect security incidents and unusual behavior.

The above mentioned techniques will be demonstrated on analyzing CoAP traffic. In our implementation, CoAP flow records include the following L7 header data: *CoAP Version*, *CoAP Message ID*, *CoAP Code* (type of the packet), *CoAP Options Count* (number of options in the header), *CoAP Content Format*, *CoAP Token*, *CoAP URI Host*, *CoAP URI Path*, and *CoAP URI Query*.

4.1 Statistical-Based Flow Analysis and Reporting

IoT flow records comprise a valuable source of statistics that describe IoT communication during a given period. Table 1 shows an example of CoAP flow records stored at the collector[7]. Each flow includes IP source and destination addresses, CoAP packet type (Code), message ID and an identifier of an IoT resource using URI host and URI path as defined by CoAP standard [35].

Table 1: Example of CoAP flows.

SrcIP	DstIP	Code	Token	MsgId	UriHost	UriPath
203.253.250.32	192.168.165.65	Put	B38A4E2E00000000	27504	192.168.165.65:15684	/3312/0/5850
192.168.165.65	203.253.250.32	Content	B38A4E1900000000	20052		
192.168.165.65	203.253.250.32	Changed	B38A4E2E00000000	27504		
192.168.165.65	203.253.250.32	Changed	B38A4E3700000000	27513		
203.253.250.32	192.168.165.65	Put	B38A4E5100000000	27539	192.168.165.65:15684	/3312/0/5850
203.253.250.32	192.168.165.65	Put	B38A4E5200000000	27540	192.168.165.65:15684	/3312/0/5850
203.253.250.32	192.168.165.60	Get	B38A4E2100000000	27491	192.168.165.60:15684	/3201/5/5550
192.168.165.60	192.168.165.62	Get	B38A4E3800000000	27514	192.168.165.62:5683	/3311/0/5706
192.168.165.60	192.168.165.62	Put	B38A4E1E00000000	27488	192.168.165.62:5683	/3311/1/5850
192.168.165.60	203.253.250.32	Server Error	B38A4E3A00000000	27516		
192.168.165.61	192.168.165.60	Content	B38A4E1300000000	973		
192.168.165.60	203.253.250.32	Changed	B38A4E2400000000	27494		
192.168.165.60	192.168.165.62	Put	B38A4E3B00000000	27517	192.168.165.62:5683	/3311/0/5706
203.253.250.34	203.253.250.32	Changed	B38A4E2C00000000	27502		

By analyzing IoT flow records we can see what CoAP requests were sent to which IoT resource, what CoAP nodes where involved in communication, when a given IoT resource was updated, etc. This data can be used to build *a baseline* that describes typical behavior

[7]For space restriction, we show only selected fields. IP statistics, timestamps and other fields are not displayed.

of the network. IoT network baseline includes a list of available IoT resources, stations that regularly communicate with these resources, a list of transmitted requests to IoT resources within a given period, peak and average network utilization, etc. By observing ongoing IoT flow, we can identify significant deviations from the baseline which may indicate misbehavior or a security incident on the network. Since IoT communication is stable in terms of connected resources and regularity of the traffic, application of baseline monitoring is very useful.

Due to the compact representation of flow records, the IPFIX collector keeps statistics of IoT communication for past few months. Using IoT flow records, we can get, for example, a list of the most requested IoT nodes, see Table 2.

Table 2: Flow Statistics: Top Talkers.

UriHost	UriPath	Bytes	Pakets
192.168.10.107:5683	/floor_1_light	179 879	4 020
192.168.10.107:5683	/floor_1_temp	162 541	4 012
192.168.10.107:5683	/.well-known/core	723	5
192.168.10.102:5683	/floor_1_light	162	3

The table shows what IoT resources identified by `UriHost` and `UriPath` values were requested during the observation period. We can see, that a light sensor identified by UriPath `/floor_1_light` and running on host 192.168.10.107 received 4.020 CoAP packets with 179.879 bytes within the given period.

By querying the flow database we can learn what stations have been sending data to that IoT node and what types of commands were used, see Table 3. We see that majority of requests were PUT commands sent by node 192.168.10.105.

Table 3: CoAP commands received by IoT node /floor_1_light.

SrcIP	Cmd	UriHost	UriPath	Bytes	Pakets
192.168.10.105	PUT	192.168.10.107:5683	/floor_1_light	4 004	179 179
192.168.10.106	PUT	192.168.10.107:5683	/floor_1_light	10	413
192.168.10.106	GET	192.168.10.107:5683	/floor_1_light	6	287

As seen above, flow records can be queried by the network administrator and presented at the *Dashboard* of the network management system where they provide on-line visibility of IoT communication in the network.

4.1.1 Whitelisting. Various studies of IoT networks and Machine to Machine (M2M) communication [4, 29] show that these networks are relatively stable in terms of number of connected devices and available resources. Thus, by running IoT flow monitoring for a certain period, e.g., one week, we can obtain a list of all available IoT resources and legitimate active nodes in the LAN. The list can be used as a source for *whitelisting* which is a filtering technique where only explicitly defined entities may access certain resources. The technique is largely used in industrial IoT networks to protect their resources [5, 30], and it is also recommended for application systems, e.g., software libraries, configuration files, etc. [34].

In case of IoT monitoring data, following items learnt during the baseline observation can be whitelisted:

- IoT nodes authorized to send/receive monitoring data.
- IoT resources authorized to be requested, updated or deleted.
- Remote hosts with permission to access IoT nodes.

IoT whitelists can be implemented using a stateful firewall where access to CoAP nodes is authorized only for previously learnt stations. Figure 4 shows an example of the Access Control List (ACL) that permits only CoAP traffic initiated by hosts that were identified during baseline observation and are listed in Table 3.

```
Router(conf)# access-list 101 remark Whitelisting CoAP traffic
Router(conf)# access-list 101 permit udp host 192.168.10.105 host 192.168.10.107 eq 5683
Router(conf)# access-list 101 permit udp host 192.168.10.106 host 192.168.10.107 eq 5683
Router(conf)# access-list 101 deny udp any host 192.168.10.107 eq 5683
Router(conf)# access-list 101 permit ip any any
```

Figure 4: Implementation of the whitelist using the ACL.

Whitelisting is a powerful technique to protect IoT resources against attackers. However, whitelists require manual updates in case of changed topology or adding new IoT nodes.

4.2 Statistical Anomaly Detection

Up to now we queried IoT flow database to get statistics about IoT communication in the network. This approach is useful for analyzing a specific security issue, e.g., observing behavior of a specific IoT node in the given time. For long-term security monitoring, we need another technique that detects anomalies in the network without manual intervention of the network administrator.

The statistical approach represents one of the possible methods for anomaly detection. In this section, we demonstrate a simple statistical method for discrimination of CoAP traffic and show that it gives promising results. This is mainly because of regularity of IoT communication. The method employs CoAP specific information obtained from CoAP-enabled IPFIX records. The presented method combines the ideas used in [32] and [14] for IP traffic fingeprinting and classification.

Assume that regular IoT traffic have some characteristics in terms of a number of packets sent, timing and amount of data exchanged. Barbosa et al. [6] even consider the traffic periodicity as the only source of information to build the traffic model. Our method creates a statistical traffic model for CoAP operation identified in the network. The model was demonstrated for two input parameters, the number of messages and the amount of data, within a fixed time period. Though the model is quite simple, it can reliably describe normal behavior and detect significant deviations. To hide his activities, the attacker would need to perform communication which does not violate the characteristics of normal communication. The model can be further refined to include other statistical parameters, such as maximum and minimum packet size, interpacket delay distribution, etc. Because the model is created for each CoAP resource and the operation that uses the resource, it is possible to identify specific patterns related to resource usage. Thus it is possible to identify situations when an attacker uses patterns that are considered normal but they were not seen before for the specific resource. On the other hand, the method relies on the possibility to inspect CoAP header fields, thus it cannot be applied to encrypted communication provided by DTLS.

4.2.1 Modeling CoAP Traffic. The proposed method monitors CoAP communication and learns patterns of usage of CoAP resource. The patterns create a *system-wide profile*. When a client wants to operate the resource, it sends a resource operation message to the server specifying its resource address. The *resource usage model* $\mathcal{M}$ thus relates an operation r^{op} on the resource r^{uri} to a statistical representation of the underlying network communication. The statistical information of model $\mathcal{M}$ is characterized by random variables $X_1, \ldots, X_k$. Each random variable X_i represents an assumption on the value of the corresponding feature. Currently, we use only two random variables: X_1 expresses a number of packets and X_2 expresses a number of octets associated with the resource operation messages[8]. The model characterizes usage of a monitored resource within the specific period of observation, which is given by a fixed size time window.

The *usage profile* $\mathcal{P}$ is created during a learning phase. It consists of a collection of resource usage models. The learner builds usage models for all observed resources using the following steps:

(1) Regular network communication is captured during the fixed interval of time. To create a model we consider a set of time windows $(w_1, \ldots, w_t)$.
(2) In each time window w_i, CoAP flows are grouped according to resource usage label $r = (r^{op}, r^{uri})$.
(3) From a set of (aggregated) flows $\{e_1, \ldots, e_m\}$ belonging to the same resource usage r a set of feature vectors $\mathcal{L} = \{\vec{x_1}, \ldots, \vec{x_m}\}$ is extracted. Each feature vector $\vec{x_i}$ statistically characterize flow e_i.
(4) The set of samples is fed to a distribution fitting procedure that applies expectation-maximization (EM) algorithm[15] to produce a mixture of normal distributions describing the sample values. The model is given as a joint probability function and a computed threshold value:

$$\mathcal{M}_i = (f_{X_1,\ldots,X_k}(x_1, \ldots, x_k), t).$$

The *threshold value* helps to distinguish whether an observed pattern is significantly different from the model of normal behavior. Let $P = \{p_i | p_i = f_{X_1,\ldots,X_k}(\vec{x_i}), (\vec{x_i}) \in \mathcal{L}\}$ is a set of probabilities of occurrences of individuals from the set of features $\mathcal{L}$ used to build the model. The threshold value of model $\mathcal{M}_i$ is given as $t = \mu(P) + x \cdot \sigma(P)$ where $\mu(P)$ is mean value of set P and $\sigma(P)$ is standard deviation. Multiplier x is used to tune the sensitivity of the model. Intuitively, threshold represents the least probability for a sample to be considered as normal behavior.

Fig. 5 shows an example of the model represented by its joint probability function. The model was created for resource usage r = (`Put`, `/floor_1_light`) where `Put` is resource operation, and `/floor_1_type` is resource URI. For creating the model in Fig. 5, 40 samples (time windows) were used to fit a mixture of normal distributions during learning phase.

4.2.2 CoAP Traffic Discrimination. Traffic flow discrimination is based on computing the similarity of currently observed behavior to the known behavior represented by corresponding resource usage model $\mathcal{M}$. In the operational phase, the network communication is analyzed as follows:

[8]It is possible to consider other features that can be extracted from IPFIX records, such as bitrate, duration, etc.

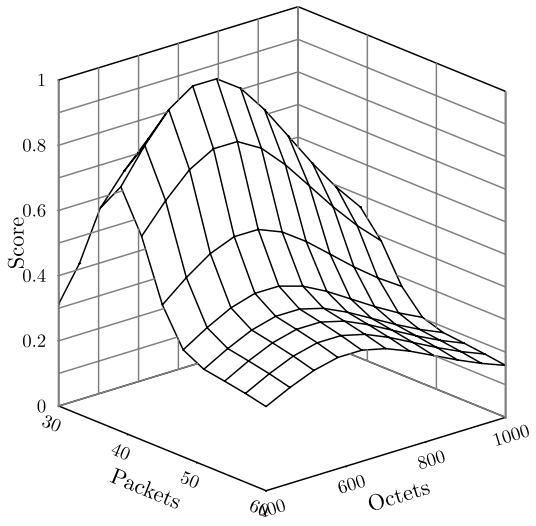

Figure 5: An example of joint probability function of sample resource usage model

(1) The traffic is captured and collected in the time window which size corresponds to that used in the learning phase.
(2) For each flow e occurred within the actual time window its resource usage label r is determined.
(3) The expected resource usage model

$$\mathcal{M} = (f_{X_1,\ldots,X_k}(x_1, \ldots, x_k), t)$$

is retrieved from the usage profile $\mathcal{P}$.
(4) Flow features are extracted from flow e to form a vector of observations $\vec{y}$.
(5) Finally, the joint probability function of model $\mathcal{M}$ is used to compute the probability for the observed behavior, $p = f_{X_1,\ldots,X_k}(\vec{y})$. If the probability is greater than threshold value t, the flow is marked as *normal*. Otherwise it is labeled as *abnormal*.

This simple procedure labels observed flows as normal or abnormal. Note that other cases not considered in the described approach can occur. It is possible that we observe a new resource usage r' not included in the profile. Such flow can be either considered as *abnormal* as it does not correspond to any known profile. Possibly, it can also be classified as *unknown* indicating that our usage profile $\mathcal{P}$ is incomplete not covering all available resources within the network.

4.2.3 Preliminary Evaluation. For the preliminary evaluation, we used four datasets of CoAP communication stored in four files, see Table 4. Files denoted as `idle`, `regular` and `observe` contain network communication reflecting regular network operations. A file denoted as `attack` contains a flooding attack executing against CoAP resources. We use all files for creating four profiles. To create a profile, we take first 20% of the source data as the training set. The testing set thus consists of remaining 80% of communication from the same file and all data from the other three files. For each executed test, the following quantities were measured:

Table 4: Datasets used for experiments

File	Packets	Flows	Resources	Normal
idle	25 096	716	5	yes
regular	54 634	1 307	12	yes
observe	17 480	415	8	yes
attack	38 474	870	8	no

Table 5: Performance measures of basic profiles

Profile	H_r	F_p
idle	39.91%	2.31 %
regular	75.98%	7.23 %
observe	84.82%	8.17 %
idle+regular	88.72%	6.36 %
idle+regular+observe	90.85%	6.46 %

- N - a number of all normal flows in the testing dataset
- N_+ - flows correctly classified as normal (true positives)
- N_- - flows incorrectly classified as normal (false positives)

The presented quantities are used to compute two metrics. Hit ratio H_r (aka recall) gives the proportion of correctly identified normal flows to all normal flows in the testing set [23]:

$$H_r = \frac{N_+}{N}$$

Hit ratio thus evaluates how good the classifier is in recognizing normal flows. The second metric is the false-positive ratio F_p (aka false discovery rate) that represents the fraction of flows incorrectly considered as normal to the total number of flows classified as normal:

$$F_p = \frac{N_-}{N_- + N_+}$$

The false-positive ratio is an important measure that tells us the degree of mistakes produced by the classifier. For experiments, it is assumed that flows in `idle`, `regular`, and `observe` data files represent normal behavior as they were all obtained by monitoring the regular operations of the CoAP system. Samples with abnormal behavior were picked up from file `attack`.

Table 5 presents results of flow classification using different profiles. The profile computed from `idle` has poor hit ratio. It is because that dataset does not contain enough patterns to represent behavior of the system adequately. We also tested profiles based on the samples drawn from multiple data files achieving hit ratio around 90%. The false-positive ratio is around 8% for the worst case. Preliminary experimental analysis shows that statistical modeling is a promising technique for profiling regular communication in IoT-based networks. The technique presented above uses an uncomplicated fully automated procedure to fit the model to regular CoAP communication and use it for discriminating unknown traffic.

5 IMPLEMENTATION

Implementation of the proposed monitoring model includes IoT protocol parsers and plugins to the IPFIX probe. Developed parsers for CoAP and LwM2M IoT protocols are available at the project web site[9]. The parsers were integrated into Flowmon Probe[10] and tested on datasets obtained from smart home testbeds at Hallym University, Chuncheon, South Korea and at Brno University of Technology, Czech Republic. Some of the datasets are available at GitHub[11]. IoT data were analyzed using a proof-of-concept implementation of anomaly detector developed by the authors.

5.1 Privacy and Security

The proposed monitoring model follows acknowledged principles of network monitoring that preserve user privacy as required by Data Retention legislation and EU General Data Protection Regulation. Monitoring data that are retrieved from IoT packets headers does not contain any personal information. On the other hand, long term monitoring of IoT devices in the smart home can disclose daily habits of the inhabitants as noticed in [13]. For this reason, users of IoT networks should be aware of monitoring and express their agreement with monitoring. For public and commercial smart buildings, privacy issues of IoT monitoring are subject to internal security regulations that must comply with the local legislation. In case of smart home monitored by an ISP, the customer should be informed about the collected data and asked to give explicit permission for IoT monitoring.

IoT flow records should be transmitted between the probe and the collector over dedicated links and encrypted in order to prevent interception or spoofing. This can be implemented by a VLAN reserved for transmitting monitoring data.

Some IoT protocols provides encryption and authentication. In case of encryption using SSL/TLS, IPSEC or Layer 7 encryption, the proposed monitoring model will not be working properly since interesting IoT headers will not be accessible by the monitoring probe any more. In this case, IoT flow monitoring is limited to detection of communicating nodes, observation of transmitted data volumes and frequency analysis of IoT transmissions as discussed in [18] and [41].

5.2 Scalability

The proposed monitoring model is focused on IoT communication that is stable, limited in computational resources and vulnerable to interception, resource scanning and command injection. It can be apply to any IoT protocols like CoAP, MQTT, LwM2M or proprietary protocols. Each IoT protocol demands a parser that will extract interesting data from IoT packet headers and build IoT flow records in the monitoring probe. No changes are required on the collector side since IPFIX provides flexible definition of IPFIX records using templates.

For IoT protocols encapsulated directly over Layer 1 and Layer 2 as mentioned in Section 3.1, a special probe with access to Layer 1 and 2 has to be implemented. In order to create full IoT flow records with L3 header data, specific-purpose IP addresses shell be selected and added to the IoT records.

This paper presents a simple IoT topology with only one IoT-enabled IPFIX probe. However, in large IoT networks, multiple

[9] See http://www.fit.vutbr.cz/research/grants/index.php.en?id=1101 [May 2019]
[10] See article https://www.flowmon.com/en/blog/flowmon-10-0-where-the-revolution-begins [May 2019]
[11] See https://github.com/matousp/IoT-related-datasets [May 2019]

probes can be deployed both in LAN segments or on the edge of the network. Each probe creates IoT flow records that are sent to the central IPFIX collector. As the IPFIX probe greatly reduces the amount of information that needs to be preserved it is suitable for monitoring of high-speed networks. It was shown that a number of flows generated are around 1/2000 of packets for campus networks [26]. Although IoT networks exhibit a different characteristic of network traffic, still the reduction ratio is significantly large to enable scalability of IPFIX monitoring system.

6 CONCLUSIONS

This paper proposes a new model for security monitoring of IoT networks. The aim of this model is to incorporate IoT flow monitoring into the current network management systems in order to increase visibility of IoT networks and to detect attacks on IoT devices. The proposed monitoring model is based on IPFIX framework as defined by RFC 7011–7013 [11, 12, 38] where the monitoring probe processes ongoing IoT packets and creates IoT flow records that are then analyzed on the IPFIX collector. We see the following advantages of this approach: IoT flow monitoring can be easily integrated into current network management systems, IoT flow records provide requested visibility of IoT communication in the network, aggregation of same IoT packets into one flow reduces number of monitored items. Unlike traditional monitoring approaches, proposed IoT flow monitoring reveals details about transmitted commands, resources, etc.

Obtained monitoring data becomes the subject to further analysis. We introduced two analysis techniques that employ IoT flow records: (i) *statistical-based flow analysis and reporting* that retrieves various statistics from IoT flow database and present them to the network administrator in the form of dashboard charts describing a big picture of active IoT communication on the network, and (ii) *statistical anomaly detection* that creates for each available IoT resource so called *resource usage model* that represents communication of this resource in terms of number of transmitted packets and octets related to the resource. Based on the previously learnt usage profiles, the method computes similarity of the ongoing traffic with the known profiles. If the probability is greater than threshold, the flow is classified as *normal*, otherwise it is marked as *abnormal*. Preliminary experiments with CoAP traffic show that the method gives promising results for discriminating irregular (abnormal) communication in the network traffic in IoT networks.

6.1 Future Work

Next work will focus on three directions. The first is implementation of IoT monitoring probe for Layer 1 and Layer 2 environment where TCP/IP stack is not implemented. This can be useful not only for smart building IoT installations but also for wide-spread global IoT networks like LoRa[12] or Sigfox[13] where network monitoring is missing.

The second direction of our research aims at development and testing of additional anomaly detection techniques in the IoT context. We plan to explore well-established approaches for anomaly detection as used in traditional IP networks and apply them to the IoT environment. This includes not only home IoT networks but also industrial IoT networks (aka SCADA systems).

The third direction is towards detailed evaluation and comparison of the IPFIX-based monitoring with other existing techniques. The direct comparison will be made to solutions that perform online packet level analysis using deep packet inspection techniques. These methods have been traditionally implemented by Intrusion Detection Systems (Snort, Bro). Also, novel approaches that use the machine learning-based methods for anomaly detection will be considered.

[12]See https://lora-alliance.org/ [May 2019]
[13]See https://www.sigfox.com/en [May 2019]

ACKNOWLEDGMENT

This work is supported by BUT project "ICT Tools, Methods and Technologies for Smart Cities" (2017–2019), no. FIT-S-17-3964, and project "IRONSTONE: IoT Monitoring and Forensics" (2016–2019), no. TF03000029, funded by the Technological Agency of the Czech Republic. The authors of the paper would like to express thanks to BUT student Patrik Krajč for his contribution on emulating CoAP network and creating CoAP datasets.

REFERENCES

[1] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. 2016. A Survey of Network Anomaly Detection Techniques. *J. Netw. Comput. Appl.* 60, C (Jan. 2016), 19–31.
[2] Pedram Amini, Reza Azmi, and MuhammadAmin Araghizadeh. 2014. Botnet Detection using NetFlow and Clustering. *Advances in Computer Science : an International Journal* 3, 2 (2014), 139–149. http://www.acsij.org/acsij/article/view/224
[3] Manos Antonakakis et al. 2017. Understanding the Mirai Botnet. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 1093–1110.
[4] R. R. R. Barbosa, R. Sadre, and A. Pras. 2012. A first look into SCADA network traffic. In *2012 IEEE Network Operations and Management Symposium*. 518–521. https://doi.org/10.1109/NOMS.2012.6211945
[5] Rafael Ramos Regis Barbosa, Ramin Sadre, and Aiko Pras. 2013. Flow whitelisting in SCADA networks. *International Journal of Critical Infrastructure Protection* 6, 3 (2013), 150 – 158. https://doi.org/10.1016/j.ijcip.2013.08.003
[6] Rafael Ramos Regis Barbosa, Ramin Sadre, and Aiko Pras. 2016. Exploiting traffic periodicity in industrial control networks. *International Journal of Critical Infrastructure Protection* (2016). https://doi.org/10.1016/j.ijcip.2016.02.004
[7] Václav Bartoš. 2015. Using Application-Aware Flow Monitoring for SIP Fraud Detection. In *Intelligent Mechanisms for Network Configuration and Security (LNCS 9122)*. Springer International Publishing, 87–99. https://doi.org/10.1007/978-3-319-20034-7_10
[8] E. Bertino and N. Islam. 2017. Botnets and Internet of Things Security. *Computer* 50, 2 (Feb 2017), 76–79. https://doi.org/10.1109/MC.2017.62
[9] J. Case, M. Fedor, M. Schoffstall, and J. Davin. 1990. A Simple Network Management Protocol (SNMP). IETF RFC 1157.
[10] B. Claise. 2004. Cisco Systems NetFlow Services Export Version 9. IETF RFC 3954.
[11] B. Claise and B. Trammel. 2013. Information Model for IP Flow Information Export (IPFIX). IETF RFC 7012.
[12] B. Claise, B. Trammel, and P. Aitken. 2013. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. IETF RFC 7011.
[13] B. Copos, K. Levitt, M. Bishop, and J. Rowe. 2016. Is Anybody Home? Inferring Activity From Smart Home Network Traffic. In *2016 IEEE Security and Privacy Workshops (SPW)*. 245–251. https://doi.org/10.1109/SPW.2016.48
[14] Manuel Crotti, Maurizio Dusi, Francesco Gringoli, and Luca Salgarelli. 2007. Traffic Classification Through Simple Statistical Fingerprinting. *SIGCOMM Comput. Commun. Rev.* 37, 1 (Jan. 2007), 5–16. https://doi.org/10.1145/1198255.1198257
[15] A.P. P. Dempster, N.M. Laird, D.B. Rubin, and D.B. Rubin. 1977. *Maximum Likelihood from Incomplete Data via the EM Algorithm.* https://doi.org/10.2307/2984875 arXiv:0710.5696v2
[16] Luca Deri, Maurizio Martinelli, and Alfredo Cardigliano. 2014. Realtime High-Speed Network Traffic Monitoring Using ntopng. In *28th Large Installation System Administration Conference (LISA14)*. USENIX Association, Seattle, WA, 78–88. https://www.usenix.org/conference/lisa14/conference-program/presentation/deri-luca

[17] Eleazar Eskin. 2000. Anomaly Detection over Noisy Data using Learned Probability Distributions. In *In Proceedings of the International Conference on Machine Learning*. https://doi.org/10.1007/978-1-60327-563-7

[18] Godred Fairhurst, Mirja Kuehlewind, Brian Trammell, and Tobias Buehler. 2018. *Challenges in Network Management of Encrypted Traffic.* WorkingPaper. ArXiv.

[19] R. Gerhards. 2009. The Syslog Protocol. IETF RFC 5424.

[20] Niv Goldenberg and Avishai Wool. 2013. Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems. *International Journal of Critical Infrastructure Protection* (2013). https://doi.org/10.1016/j.ijcip.2013.05.001

[21] Mark Graham, Adrian Winckles, and Erika Sanchez-Velazquez. 2015. Practical Experiences of Building an IPFIX Based Open Source Botnet Detector. *The Journal on CyberCrime & Digital Investigations* (12 2015), 21–28.

[22] Jorge Granjal, João M. Silva, and Nuno Lourenço. 2018. Intrusion detection and prevention in CoAP wireless sensor networks using anomaly detection. *Sensors (Switzerland)* 18, 8 (2018). https://doi.org/10.3390/s18082445

[23] Jiawei Han, Micheline Kamber, and Jian Pei. 2011. *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[24] David Hanes, Gonzalo Salqueiro, Patrick Grossetete, Rob Barton, and Jereme Henry. 2017. *IoT Fundamentals. Networking Technologies, Protocol and Use Cases for the Internet of Things.* Cisco Press.

[25] R. Hofstede, V. BartoÃź, A. Sperotto, and A. Pras. 2013. Towards real-time intrusion detection for NetFlow and IPFIX. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*. 227–234. https://doi.org/10.1109/CNSM.2013.6727841

[26] Rick Hofstede, Pavel Čeleda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. 2014. Flow monitoring explained: From packet capture to data analysis with NetFlow and IPFIX. *IEEE Communications Surveys and Tutorials* (2014). https://doi.org/10.1109/COMST.2014.2321898

[27] Rick Hofstede, Pavel Čeleda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. 2014. Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE COMMUNICATIONS SURVEYS AND TUTORIALS* 16 (2014). https://doi.org/10.1109/COMST.2014.2321898

[28] IEC. 2016. *Information technology – Message Queuing Telemetry Transport (MQTT).* Standard ISO/IEC 20922:2016. International Organization for Standardization.

[29] S. S. Jung, D. Formby, C. Day, and R. Beyah. 2014. A first look at machine-to-machine power grid network traffic. In *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 884–889. https://doi.org/10.1109/SmartGridComm.2014.7007760

[30] Eric D. Knapp and Joel Thomas Langill. 2015. *Industrial Network Security. Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems.* Syngress.

[31] Federico Maggi and Rainer Vosseler. 2018. *The Fragility of Industrial IoT's Data Backbone. Security and Privacy Issues in MQTT and CoAP Protocols.* Technical Report. Trend Micro. https://documents.trendmicro.com/assets/white_papers/wp-the-fragility-of-industrial-IoTs-data-backbone.pdf

[32] Constantine Manikopoulos and Symeon Papavassiliou. 2002. Network intrusion and fault detection: A statistical anomaly approach. *IEEE Communications Magazine* (2002). https://doi.org/10.1109/MCOM.2002.1039860

[33] Omar Santos. 2016. *Network Security with NetFlow and IPFIX. Big Data Analytics for Information Security.* Cisco Press.

[34] Adam Sedgewick, Murugiah Souppaya, and Karen Scarfone. 2015. *Guide to Application Whitelisting.* Technical Report NIST SP-800-167. National Institute of Standards and Technology.

[35] Z. Shelby, K. Hartke, and C. Bromann. 2014. The Constrained Application Protocol (CoAP). IETF RFC 7252.

[36] Federico Simmross-Wattenberg, Juan Ignacio Asensio-Pérez, Pablo Casaseca-De-La-Higuera, Marcos Martín-Fernandez, Ioannis A. Dimitriadis, and Carlos Alberola-López. 2011. Anomaly detection in network traffic based on statistical inference and α-stable modeling. *IEEE Transactions on Dependable and Secure Computing* 8, 4 (2011), 494–509. https://doi.org/10.1109/TDSC.2011.14

[37] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani. 2015. Network-level security and privacy control for smart-home IoT devices. In *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 163–167. https://doi.org/10.1109/WiMOB.2015.7347956

[38] B. Trammel and B. Claise. 2013. Guidelines for Authors and Reviewers of IP Flow Information Export (IPFIX) Information Elements. IETF RFC 7013.

[39] B. Trammell and E. Boschi. 2011. An introduction to IP flow information export (IPFIX). *IEEE Communications Magazine* 49, 4 (April 2011), 89–95. https://doi.org/10.1109/MCOM.2011.5741152

[40] D. Van der Steeg, R. Hofstede, A. Sperotto, and A. Pras. 2015. Real-time DDoS attack detection for Cisco IOS using NetFlow. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. 972–977. https://doi.org/10.1109/INM.2015.7140420

[41] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar. 2015. A Survey of Methods for Encrypted Traffic Classification and Analysis. *Netw.* 25, 5 (Sept. 2015), 355–374. https://doi.org/10.1002/nem.1901