Model-Driven Network Monitoring Using NetFlow Applied to Threat Detection

Daniel González-Sánchez
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Madrid, Spain
0000-0002-7691-0030

Ignacio D. Martinez-Casanueva
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Madrid, Spain
0000-0002-8573-127X

Antonio Pastor
Global CTIO
Telefónica Investigación y Desarrollo
Madrid, Spain
0000-0003-2849-9782

Luis Bellido Triana
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Madrid, Spain
0000-0001-9591-0928

Cristina Pinar Muñoz Zamarro
Global CTIO
Telefónica Investigación y Desarrollo
Madrid, Spain
0000-0003-2023-8967

Alejandro Antonio Moreno Sancho
Global CTIO
Telefónica Investigación y Desarrollo
Madrid, Spain
0000-0001-8863-1875

David Fernández Cambronero
Dpto. de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Madrid, Spain
0000-0002-2172-9162

Diego Lopez
Global CTIO
Telefónica Investigación y Desarrollo
Seville, Spain
0000-0002-8326-2000

***Abstract*—In recent years, several research works have proposed the analysis of network flow information using machine learning in order to detect threats or anomalous activities. In this sense, NetFlow-based systems stand out as one of the main sources of network flow information. In these systems, NetFlow collectors provide the flow monitoring information to be analyzed, but the particular information structure and format provided by different collector implementations is a recurring problem. In this paper, a new YANG data model is proposed as a standard model to use NetFlow-based monitoring data. In order to validate the proposal, a NetFlow collector incorporating the proposed NetFlow YANG model has been developed, to be integrated in a network scenario in which network flows are analyzed to detect malicious cryptomining activity. This collector extends an existing one, and provides design patterns to incorporate other existing collectors into this common data model. Our results show how, by using the YANG modeling language, network flow information can be handled and aggregated in a formal and unified way that provides flexibility and facilitates data analysis applied to threat detection.**

***Index Terms*—NetFlow, YANG, data model, threat detection, cryptomining, network monitoring.**

I. Introduction

Historically, network flows have been considered as a key source of information for network security analysis, with the objective of detecting anomalous activities that traditional security systems cannot handle [1]. In recent years, new approaches have applied machine learning analysis on network flow information to detect threats or vulnerabilities in a network. One of the recently reported results is based on the application of machine and deep learning models for the detection of cryptomining flows in real time, offering a solution to industry for detecting cryptojacking activity over the network [2].

NetFlow is one of the main protocols to provide flow-level network measurements and one of the most used in threat detection. However, there are different implementations of NetFlow collectors that provide the flow monitoring information using their own particular structure and format. This means that there is no standard data model that an application can use when trying to consume NetFlow monitoring data. Not having a unified data model makes it difficult to decouple data analysis and collection, making analytics tools heavily dependent on the specific format, usually based on CSV. While a CSV format can be considered simple and adequate for basic analysis, it does not support a structured schema suitable for applying a deep data science analysis, as it lacks mechanisms to identify and validate data patterns and associated metadata.

To solve this problem and provide a standardized data model, we propose a YANG data model that NetFlow collectors can leverage to export NetFlow-based monitoring data in a formal and unified fashion. This will also enable the aggregation of network flows in a common way that facilitates the application of machine learning mechanisms applied to threat detection such as cryptomining attacks.

The remainder of the paper is organized as follows. Section II provides insight on research activity related to the use of NetFlow in threat detection. Also, this section provides a review of available NetFlow collector implementations, as well as determining if there are available implementations of

YANG models about NetFlow. Section III describes the details about the YANG model proposed for NetFlow collectors. Section IV describes a proposal for a NetFlow collector based on the YANG model. Section V presents a use case that shows how a threat detection system benefits from NetFlow-based monitoring data modeled by YANG. Finally, Section VI provides the conclusions and future steps of this research.

II. State of the Art

A. NetFlow for Threat Detection

The generic concept of network flow, where all packets related to a specific communication application are grouped together, has been considered a valuable tool to scale in the identification and monitoring of network activity. How to deliver flow-related information from network devices to management systems was addressed with a set of proprietary protocols, where NetFlow was the reference and was added to IETF informational standards in its version 9 (RFC 3954) [3], and later evolved to the IPFIX [4] standard. NetFlow has increased its value for security monitoring over the years. Today, NetFlow is considered a valuable source of information in threat detection, such as intrusion detection systems or network attacks, and with the advent of Machine Learning techniques it has become the reference for relevant datasets, as it is widely supported in any kind of network [5]–[7]. In [8] the authors propose to use NetFlow fields with some extensions as a standard feature set for Machine Learning evaluation and select the CSV format commonly applied in data science research.

B. NetFlow and the YANG Language

In our study, we propose to use the YANG [9] data modeling language for NetFlow. YANG is widely used today in network management solutions and standards, such as IETF NETCONF [10] or OpenConfig [11]. Using YANG will allow us to validate flows collected from the network with a standard model and to apply automated transformations.

YANG is agnostic to the data encoding format and the transport protocol. YANG is a human-friendly, extensible language rich in semantics, that structures data following a tree hierarchy into what is known as a YANG module.

In regards to NetFlow, there are YANG modules developed by Cisco with the purpose of managing NetFlow exporters in Cisco IOS-XR devices [12]. However, we have not found YANG modules related to modeling NetFlow data itself.

C. NetFlow Collectors

NetFlow version 9 specification states that a NetFlow collector is the entity responsible for receiving export packets from NetFlow exporters. Then, the contents of each export packet are processed and sent to subscribed telemetry consumers. Several implementations of NetFlow collectors can be found in the open source community.

GoFlow2 [13] is an open source collector implementation that gathers network information from different flow protocols (i.e., NetFlow, IPFIX, and sFlow), and serializes into encoding formats such as JSON or Protobuf. *GoFlow2* supports NetFlow version 9, and both IPv4 and IPv6. In addition, it can send the collected data to a message queue service such as Apache Kafka.

The *nfdump* [14] solution is a toolset for collecting and processing flow data sent from network devices. Among its main features, *nfdump* supports NetFlow version 9, IPFIX and sFlow protocols, and both IPv4 and IPv6. It has the option of providing the collected information as binary format or as *pcap* files. In addition, *nfdump* includes the option for serializing the information into CSV format for post-processing.

While the CSV format could be useful for applying basic data science analysis, it is not the proper input format for artificial intelligence engines that can be used for threat detection. In this sense, a common hierarchical data format such as JSON is more appropriate than the comma-separated values provided by CSV, as the former allows structuring the data according to a data schema where all the different values are properly identified and ordered. Having the data in such a structured format makes it easier and more efficient to handle and process large volumes of data in order to detect anomalous patterns in the network.

III. A YANG Data Model for NetFlow Collectors

A new YANG module aligned with NetFlow version 9 has been developed [15]. This module captures the contents from the NetFlow export packet that are relevant for subscribers to NetFlow monitoring data: packet header and data flow sets. The resulting YANG module as depicted in Fig.1 comprises two main YANG containers: *collector* and *export-packet*.

The *collector* YANG container includes metadata related to the reception of an export packet in the NetFlow collector. These metadata range from reception timestamp to the IP address – either version 4 or version 6 – of the NetFlow exporter that produced such packet.

The *export-packet* YANG container incorporates the contents of an export packet. This container draws from the NetFlow Version 9 Flow-Record Format specification [16] which provides a thorough description of the structure of a NetFlow export packet. First, the data in the packet header are modeled using YANG leafs that represent fields such as the packet's sequence number or the identifier of the NetFlow exporter that sent the packet, i.e., source ID. Second, a YANG list named *flow-data-record* models the data pertaining to the different data flow sets found within an export packet. To uniquely identify each flow within the YANG list, the *flow-id* YANG leaf is designated the key of the list. The *flow-id* leaf represents the hash code obtained from the 5-tuple formed by the source IP address, destination IP address, source port, destination port, and protocol number.

Aside from the *flow-id* leaf, the child nodes of the *flow-data-record* container represent all possible fields that can be included in a data flow set. The data fields present in each data flow set will vary depending on the type of network traffic that has been sampled with NetFlow. Thus, the proposed YANG module introduces multiple YANG containers, each

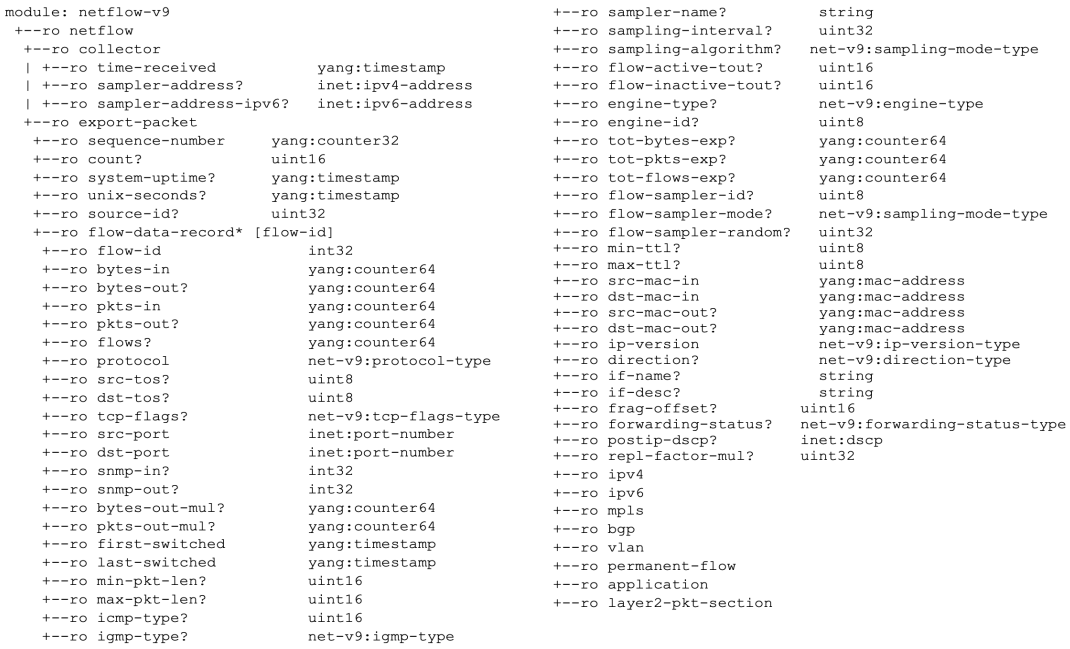

Fig. 1. YANG module for NetFlow version 9. The YANG tree representation is displayed in two columns for better readability.

enclosing those data fields that are specific to a particular network protocol. For instance, *mpls* contains a set of possible MPLS labels for each position in the stack; *bgp* holds data related to BGP such as next-hop IP address or the autonomous system (AS) numbers of the peers involved in the BGP session; whereas the *vlan* container groups the source and destination VLAN identifiers that may have been used in the sampled network flow. On the other hand, data related to the IP protocol is also grouped into a separate YANG container: one for IP version 4 and another for IP version 6. As a result, the contents of the data flow set follow a clear structure within the YANG module, hence, improving its readability and enabling other YANG modules to import these groups.

Overall, the proposed YANG module attempts to exploit the data typing functionality of the YANG language by rigorously defining all nodes in the module. In this sense, fields such as *protocol* or *igmp-type*, which are commonly modeled as integers, are defined in the module as enumerates with their respective semantic explanation. The *tcp-flags* field, which represents a list of flags that are optionally enabled in a TCP flow, is accurately defined using the *bits* type. Other data fields related to IP or MAC addresses, are modeled by importing data types from standardized YANG modules such as *ietf-inet-types* and *ietf-yang-types*.

IV. A YANG-BASED NETFLOW COLLECTOR

As noted previously, NetFlow monitoring data can be collected by different collector implementations, each of them providing the network flows in a particular encoding format. In order to deal with the encoding aspect, a special component called NetFlow Driver has been implemented. The NetFlow Driver is a piece of software that is responsible for structuring the output of a NetFlow collector into the NetFlow YANG data model, and then serializing the data using a particular encoding format. This NetFlow Driver is implemented based on the YANG Tools project provided by OpenDayLight [17]. YANG Tools is a set of libraries and tooling that supports the use of YANG in Java programming language, and allows normalizing data according to a specific YANG data model and serializing that YANG-modeled data into a JSON or XML encoding format. For the sake of interoperability, we will use the JSON-IETF format, which is a standardized JSON encoding format for representing YANG-modeled data [18].

The development of the NetFlow Driver is mainly based on the Java bindings generated by YANG Tools. The Java bindings are the structuring of the data model representation in the form of Java code that are automatically generated from the defined YANG model. Using these bindings, a concrete representation of the YANG data tree can be built in order normalize data that matches with the relative YANG model.

It is important to note that the NetFlow Driver implementation is particularized to the *GoFlow2 Collector*. This NetFlow collector solution was chosen mainly because it is compatible with NetFlow version 9, as well as the specification of the fields associated with the NetFlow records it generates is fairly well aligned with the NetFlow YANG model. In addition, *GoFlow2* allows serializing NetFlow monitoring data in the JSON encoding format, which makes the data processing more efficient, easier and human-friendly in the corresponding NetFlow Driver implementation.

Fig.2 depicts the proposed reference architecture for the collection of NetFlow monitoring data modeled with YANG. First, the *NetFlow Exporter* entity is a NetFlow-enabled device that generates flow records and periodically exports them in

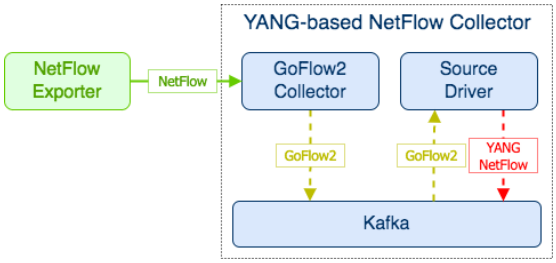

Fig. 2. A Reference Architecture for a YANG-based NetFlow Collector.

form of export packets to the NetFlow collector. Then, the *GoFlow2 Collector* is responsible for receiving the export packets, and pre-processing them in order to interpret the existing data flow sets. In addition, the *GoFlow2 Collector* stores the collected NetFlow-based data in a message queue solution based on an Kafka broker. The NetFlow monitoring data is aggregated in a particular Kafka topic following the JSON encoding format supported by the *GoFlow2* solution. Once the NetFlow monitoring data is available in Kafka, the functionality of the NetFlow Driver comes into play. The NetFlow Driver (i.e., the *Source Driver* entity in Fig.2) will be responsible for adapting the NetFlow monitoring data by parsing the raw data aggregated by the *GoFlow2 Collector* in Kafka and producing data structured according to the associated NetFlow YANG model. The normalized NetFlow data will be aggregated back in another Kafka topic.

Having the NetFlow monitoring data normalized according to the YANG model provides certain advantages for analysis related to data validation and cleansing. The YANG model works as a validation schema for the data collected from the NetFlow collector. Then, the data normalization process ensures that the name, data types, units and the range of allowed values of the different flow fields collected are correct. In addition, the order of the flow fields is not a problem because they are matched according to the YANG model that validates that each of the field exists. Thus, it would avoid the need to perform a detailed pre-processing phase on the data flows that are analyzed. Having the network flows modeled with YANG also allows reducing the possibility of obtaining measurement errors from the NetFlow collector, which in turn could reduce the appearance of false positives and false negatives when performing machine learning analysis.

V. Use Case: NetFlow Monitoring for Cryptomining Threat Detection

We have selected one representative case for threat detection: the cryptomining problem. Nowadays, cyber criminals focus on compromising computing-intensive sites and services (e.g., cloud computing systems), where they can introduce malicious software to run high resources consuming activities for mining digital coins. This activity requires network connectivity with mining pools, a set of services that distribute working loads and provide economic rewards for the activity [19]. This specific network activity can be detected using NetFlow [20]. In this section, we show a solution to provide additional performance features for NetFlow-related monitoring data by means of an augmented YANG model, which could be useful to detect cryptomining activity on the network.

A. Experimental Threat Detection System

Fig.3 depicts the full data pipeline process for network flow normalization applied to threat detection. It represents a reference architecture that enables the collection, aggregation, and delivery of NetFlow-based monitoring data modeled on YANG. This architecture is a experimental system in which to analyze the benefits that having NetFlow data normalized according to a YANG model bring to Artificial Intelligence (AI) engines that process network flows.

As described in the previous section, the NetFlow monitoring data is normalized and aggregated following a formal data model thanks to the *YANG-based NetFlow Collector*. By applying a formal YANG model that is agnostic to the the NetFlow collector, the system provides interoperability to prospective consumers. In our proposal, we consider a particular use case where AI engines consume and process network flows to detect threats related to malicious cryptomining activity. For this use case, we consider two possible scenarios.

On the one hand, a legacy AI engine that consumes NetFlow monitoring data following a particular CSV schema to apply data science analysis. In that case, the NetFlow monitoring data structured according to the YANG model in a JSON-IETF format must be decoded into a CSV serialization format. In order to cover this decoding process, we need another type of YANG-based Driver application (i.e., *Consumer Driver*) for mapping the data to the desired CSV schema using the YANG model and the bindings provided by YANG Tools library. It demonstrates the potential of our solution to provide an abstraction from the serialization format. The *Consumer Driver* acts as a intermediary entity to establish an agreement between the input YANG model and the output CSV schema in order to adapt the NetFlow data to the format desired by the consuming AI engine. The drawback of this approach is

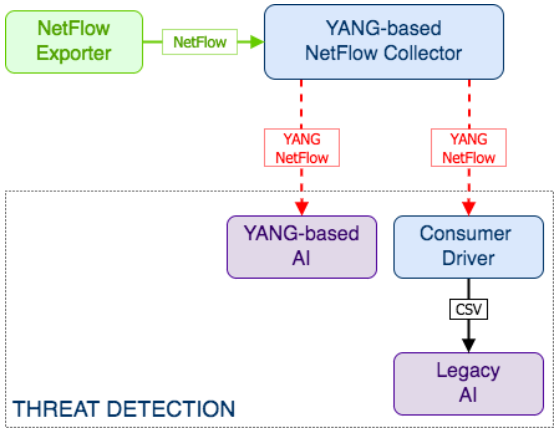

Fig. 3. NetFlow data pipeline process for threat detection.

that it depends on the desired CSV schema. Depending on flow fields considered in the CSV schema, the implementation of the *Consumer Driver* changes, since it only performs the mapping of the data fields required by the schema itself. Also, this approach does not support an extension of the input YANG model. If the YANG model evolves to include new information, the *Consumer Driver* implementation must be updated so that the information can be mapped to the consumer's data schema. It avoids a dynamic adaptation of the NetFlow data in case of evolutionary changes of the associated YANG model.

To overcome these shortcomings, an alternative approach is proposed. This approach is based on the ideal solution in which the AI engine supports the consumption of YANG-modeled data (i.e., the *YANG-based AI* entity depicted on Fig.3). In such case, the NetFlow monitoring data doesn't need to be adapted through a *Consumer Driver*. Then, once the NetFlow monitoring data is normalized according with the YANG model, it can be consumed directly by the *YANG-based AI*. In this sense, the YANG model acts as a single and dynamic agreement between the NetFlow collector and the consumer. The approach is dynamic because it successfully supports the continuous evolution of the YANG model. If the NetFlow YANG model suffers a modification, augmentation or deviation, the corresponding normalized data will be automatically updated by the *Source Driver* in order to be consumed by the *YANG-based AI engine*. In addition, this solution would facilitate performing anomaly predictions in the AI engine. One of the benefits of YANG models is that they are human-friendly and rich in semantics. Therefore, being able to directly consume network flows modeled with YANG would help to better interpret and determine those flow fields that are most appropriate for predicting network anomalies. Being able to recognize in detail the meaning and range of values supported by each flow field helps to determine the information that is useful for training a machine learning model but, at the same time, avoiding undesirable overfitting.

B. Aggregations applied to NetFlow for Threat Detection

Another advantage of supporting the evolution of NetFlow data modeled by YANG is the ability to aggregate the flow data in order to obtain additional information. In this situation, the *YANG-based AI* engine expects to receive the NetFlow monitoring data with additional parameters aggregated for each flow. In such a case, the related machine learning model requires network flows with aggregated performance information (i.e., aggregated KPIs) as input data, and an *Aggregation Application* running as a stream processing service on a data engineering engine can compute them in advance. An update version of the data pipeline is depicted in Fig.4 in order to include the aggregation process. This *Aggregation Application* performs data transformations over the normalized network flows through different operations, such as field mapping, filtering, joins, or windowed aggregations.

In this sense, the semantic information related to the new calculated KPIs must be taken into account. Therefore, an

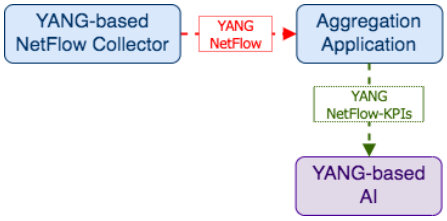

Fig. 4. NetFlow data pipeline process supporting data aggregations.

extension of the original YANG model must be defined. By means of the *augment* statement, a YANG module can insert additional nodes into the YANG model.

Fig.5 depicts an augmentation of the original YANG model by extending the list of child nodes from the *flow-data-record* container with new data nodes. The new child nodes represent aggregated KPIs computed through the *Aggregation Application* from an original data flow set collected by the NetFlow collector. The semantics of these new aggregated KPIs are summarized as follows:

- *flow-duration*: Duration of the flow. Difference between *last-switched* and *first-switched* data nodes.
- *bytes-in-per-second* and *bytes-out-per-second*: Counters for the number of incoming and outgoing bytes per second associated with an IP flow.
- *pkts-in-per-second* and *pkts-out-per-second*: Counters for the number of incoming and outgoing packets per second associated with an IP flow.
- *bytes-in-per-packet* and *bytes-out-per-packet*: Counters for the number of incoming and outgoing bytes per packet associated with an IP flow.

Then, we define a new YANG module that augments the original model and includes the aforementioned semantic information about the aggregated KPIs. It is important to note that once the aggregated KPIs are calculated, the *Aggregation Application* adapts the data according to the augmentation data model. In order to complete the adaptation process, the data is mapped through the new bindings generated by YANG Tools from the new YANG module, and then serialized into the encoding format desired by the AI engine consumer.

The aggregated KPIs provide information related to the volume of data at the flow-level. This information can be useful for the AI engine in order to detect different anomalies in the analyzed network flows. For example, let's imagine a case where there are repeated network flows of short *flow-duration* associated with the same source IP. Also, let's consider that

```
module: netflow-v9-agg

  augment /net-v9:netflow/net-v9:export-packet/net-v9:flow-data-record:
    +--ro flow-duration?          yang:timestamp
    +--ro bytes-in-per-second?    per-decimal
    +--ro bytes-out-per-second?   per-decimal
    +--ro pkts-in-per-second?     per-decimal
    +--ro pkts-out-per-second?    per-decimal
    +--ro bytes-in-per-packet?    per-decimal
    +--ro bytes-out-per-packet?   per-decimal
```

Fig. 5. NetFlow YANG model augmentation with aggregated KPIs.

the flow has a high value for the calculated *pkts-in-per-second* and *pkts-out-per-second* KPIs. These results about suspicious bursts of traffic can be processed and analyzed by the AI engine in order to predict and determine threat detection, such a DoS attempts or cryptomining activities.

Another aggregation possibility is combining data from different data sources. If each data source supports its own YANG model, the data aggregation problem is reduced to a combination of multiple YANG models. Then, this results in a new YANG data model that imports the YANG modules related to the different data sources. This solution allows combining data from monitoring sources of different purposes to perform different transformations or new KPI calculations. Applying this data aggregation process to threat detection, one possible and interesting scenario could be aggregate data related to NetFlow-based monitoring and computing resources consumption on the same network device. Creating a joint and unified data model facilitates the aggregation process to combine and correlate information in order to detect anomalies. In this case, a substantial increase in CPU consumption coupled with a burst in traffic volume could lead to the detection of malicious cryptomining activity on the monitored device.

VI. Conclusions and Future Work

In this paper, we propose a new YANG model for NetFlow-based monitoring data. This YANG model is aligned with NetFlow version 9 and captures the semantic information related to the export packets collected by NetFlow collectors. In addition, we present a reference architecture that enables the collection, aggregation, and delivery of NetFlow-based monitoring data modeled on YANG. This architecture aims to demonstrate the benefits of having NetFlow monitoring data structured according to a YANG model for AI engines that analyze network flows for threat detection, taking advantage of the semantics, flexibility, and scalability provided by the YANG models. In addition, a use case is presented where the evolutionary capability of YANG models is demonstrated by performing transformations of NetFlow monitoring data in order to calculate KPIs that are useful for the detection of anomalies such as cryptomining activities.

For future steps, we intend to carry out a practical demonstration and scalability study of the presented experimental threat detection system that will help us to fully validate the proposal. Also, we propose to leverage the versatility provided by YANG modeling to create new derived YANG models with additional data sources aggregated. One example could be a YANG model that aggregates NetFlow fields along with other resource consumption information for the same source device. This would improve the performance of the detection procedures applicable to bot-based abuse like cryptomining, since these activities are associated with high computing or storage resource usage.

Acknowledgment

The research leading to these results received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 883335 (PALANTIR) and no. 871808 (INSPIRE-5Gplus). The paper reflects only the authors' views. The Commission is not responsible for any use that may be made of the information it contains. In addition, this work was supported by the Spanish Ministry of Economy and Competitiveness and the Spanish Ministry of Science and Innovation in the context of ECTICS project under Grant PID2019–105257RB-C21 and Go2Edge under Grant RED2018–102585-T.

References

[1] B. Li, J. Springer, G. Bebis, and M. Hadi Gunes, "A survey of network flow applications," *Journal of Network and Computer Applications*, vol. 36, no. 2, pp. 567–581, 2013.

[2] A. Pastor, A. Mozo, S. Vakaruk, D. Canavese, D. R. López, L. Regano, S. Gómez-Canaval, and A. Lioy, "Detection of encrypted cryptomining malware connections with machine and deep learning," *IEEE Access*, vol. 8, pp. 158036–158055, 2020.

[3] B. Claise, "Cisco Systems NetFlow Services Export Version 9." RFC 3954, Oct. 2004.

[4] P. Aitken, B. Claise, and B. Trammell, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information." RFC 7011, Sept. 2013.

[5] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "Netflow datasets for machine learning-based network intrusion detection systems," in *Big Data Technologies and Applications* (Z. Deze, H. Huang, R. Hou, S. Rho, and N. Chilamkurti, eds.), (Cham), pp. 117–135, Springer International Publishing, 2021.

[6] H. Attak, M. Combalia, G. Gardikis, B. Gastón, L. Jacquin, D. Katsianis, A. Litke, N. Papadakis, D. Papadopoulos, A. Pastor, M. Roig, and O. Segou, "Application of distributed computing and machine learning technologies to cybersecurity," *Computer & Electronics Security Applications Rendez-vous (C&ESAR), 19-21 November 2018*, no. November, pp. 1–16, 2018.

[7] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[8] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a Standard Feature Set for Network Intrusion Detection System Datasets," *Mobile Networks and Applications*, 2021.

[9] M. Björklund, "The YANG 1.1 Data Modeling Language." RFC 7950, Aug. 2016.

[10] R. Enns, M. Björklund, A. Bierman, and J. Schönwälder, "Network Configuration Protocol (NETCONF)." RFC 6241, June 2011.

[11] "OpenConfig - Home." https://www.openconfig.net/. Accessed: 2022-03-21.

[12] "Cisco-IOS-XR-traffmon-netflow-cfg." https://yangcatalog.org/yang-search/module_details/Cisco-IOS-XR-traffmon-netflow-cfg. Accessed: 2022-03-21.

[13] "netsampler/goflow2: High performance sFlow/IPFIX/NetFlow Collector." https://github.com/netsampler/goflow2. Accessed: 2022-03-21.

[14] "phaag/nfdump: Netflow processing tools." https://github.com/phaag/nfdump. Accessed: 2022-03-21.

[15] "NetFlow v9 YANG model." https://github.com/cristinapmz/netflow-v9. Accessed: 2022-03-21.

[16] Cisco Systems, "Cisco IOS NetFlow Version 9 Flow-Record Format," *White Paper*, no. May, pp. 1–14, 2011.

[17] "YANG Tools Developer Guide — OpenDaylight Documentation." https://docs.opendaylight.org/en/latest/developer-guides/yang-tools.html. Accessed: 2022-03-21.

[18] L. Lhotka, "JSON Encoding of Data Modeled with YANG." RFC 7951, Aug. 2016.

[19] S. Pastrana and G. Suarez-Tangil, "A first look at the crypto-mining malware ecosystem: A decade of unrestricted wealth," *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 73–86, 2019.

[20] J. Z. I. Munoz, J. Suarez-Varela, and P. Barlet-Ros, "Detecting cryptocurrency miners with NetFlow/IPFIX network measurements," *2019 IEEE International Symposium on Measurements and Networking (M&N)*, 2019.