OPTIMIZATION OF DATA TRANSFER IN THE INTERNET OF THINGS ENVIRONMENT

Rastislav PETIJA, František JAKAB, Peter FECIĽAK, Miroslav MICHALKO
Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic
Tel. 055/602 7013, E-mails: rastislav.petija@tuke.sk, frantisek.jakab@tuke.sk
peter.fecilak@tuke.sk, miroslav.michalko@tuke.sk

ABSTRACT

This article deals with the implementation and experimental verification of the suitability of the TinyIPFIX protocol for data transmission in the Internet of Things environment. The work was devoted to the creation of three main components, namely TinyIPFIX exporter, collector, and mediator. The implementation of these tools made it possible to extend the possibility of monitoring a common network with an IoT environment. The experiments confirmed the success of the implementation of the protocol based on standards and pointed out the suitability of the implementation of the TinyIPFIX protocol mainly due to its optimized processes, which save up to 72% of bandwidth consumption compared to the IPFIX protocol when transmitting one data unit. Thanks to the modular approach during implementation, it is possible to deploy the protocol in the environment regardless of the transport technology. The created solution can therefore also be used in UAV sensor networks.

Keywords: *Internet of Things, data flow, IPFIX, TinyIPFIX*

1. INTRODUCTION

The Internet of Things environment is a limited environment that consists of a number of different devices whitch use different technologies and protocols [1]. These are usually simple devices that have several constraints [2] like limited amount of electricity, computing and storage resources and, last but not least, a limited amount of bandwidth for data transmission. The latter fact is mainly related to the fact that communication, which consists of either sending data or receiving data, consumes considerable energy and therefore it is necessary to keep the time required for communication to a minimum. A small amount of time to send and receive data, in turn, requires the use of communication protocols that can operate in a gentle mode. It is ideal to use protocols that do not maintain a connection.

Another problem in the IoT environment is the lack of standardization, which is because there are a large number of manufacturers of IoT devices, where each implements its own protocols, which are often not compatible with devices from other manufacturers. Since our research aims to extend the possibility of monitoring a common network to the IoT environment, it is important to have tools that can transmit data from the entire IoT environment in an efficient and standardized way. Tools should use a protocol that can transmit mainly statistics on devices, data flows and, ideally, common data in an efficient way, and should be a protocol that can be used to transmit data on common networks to interconnect various monitored domains. The analysis showed that the TinyIPFIX protocol could be considered the most suitable protocol to meet these requirements. The content of this publication is an overview of the tools we have created and experiments performed to verify the correct implementation of the TinyIPFIX protocol for IoT devices and to confirm the suitability of this protocol for the transmission of information in the IoT.

2. PROTOCOLS FOR THE TRANSMISSION OF DATA FLOW INFORMATION

Traditional networks use the proprietary NetFlow protocol and the open IPFIX standard to transmit data about data flows. TinyIPFIX is a simplified version of IPFIX for restricted environments. Both protocols are adapted to be able to transmit various statistics, which can be used to identify devices and to describe the properties and characteristics of the data streams they transmit. Most commonly, data such as source and destination IP address, source, and destination port of the OSI model transport layer, and the protocol number used are used to identify the data flow. Subsequently, the data characterizing the data flow are most often the number of transmitted packets, the number of bytes transmitted, and the duration of the communication. A detailed description of the IPFIX protocol can be found in the RFC 7011 standard [3].

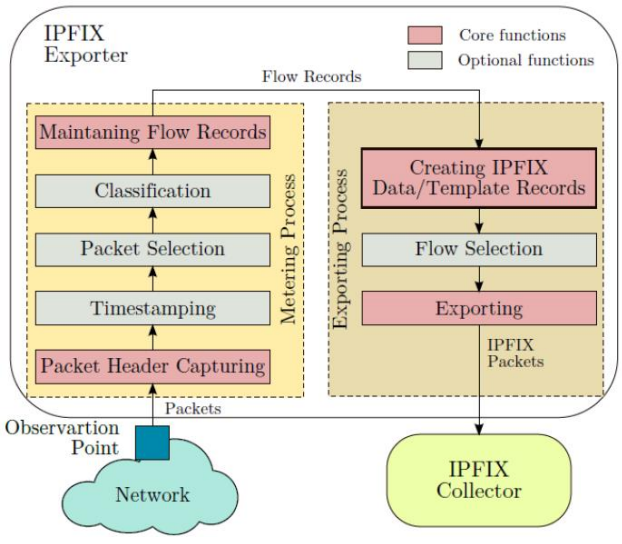

Fig. 1 General Architecture for the IPFIX exporter

Several research have been conducted on architectures for capturing data, creating data streams, and collecting qualitative data about them, while the general architecture for the IPFIX exporter can be seen in the Fig. 1 which can also be found in publications [6].

Each protocol for the transmission of data on data flows has two main parts, namely the process of export and data collection. The data export process is usually located on a limited IoT device from which the data needs to be sent and the data collection process can be found according to the architecture in several places, e.g., another IoT device, IoT gateway, or server designated for that.

The general architecture based on standard definition in RFC 8272 [4] describing the TinyIPFIX protocol can be found in Fig. 2.

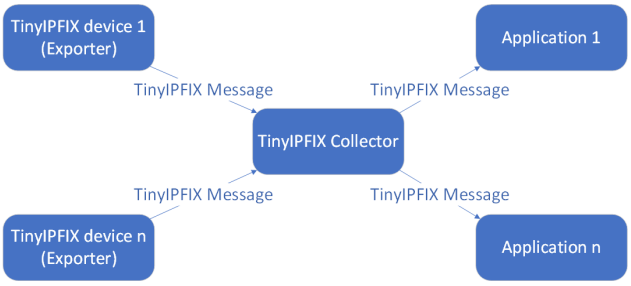

Fig. 2 General architecture of the TinyIPFIX protocol

The IPFIX and TinyIPFIX protocols also allow the implementation of a mediation process, which is a key element in enabling mutual communication between multiple domains. The mediation process allows for two main functions. The first is the transparent transfer of data from one observed domain to another without changing the format and content of the data. The second is mediation, which allows to receive data in TinyIPFIX format and send it in IPFIX format. It is the type of mediation that is key to extending the monitoring of common networks to the IoT environment.

Both protocols, IPFIX and TinyIPFXI, are used to transmit data about data streams. The protocols are based on the principle of sending templates, which define the meaning of later transmitted data and on the sending of the data itself. The collector saves the received template and interprets the following received data based on it. The only difference in the protocols is the header optimization in the TinyIPFIX protocol. The basic concept remains the same due to the mutual compatibility of protocols.

3. EXPERIMENTS WITH DATAFLOW TRANSMISSION FROM IOT ENVIRONMENT TO MONITORING SYSTEM

One of the first experiments related to the implementation of the TinyIPFIX protocol for data transmission was published in the article [7]. There are several implementations of this protocol, but they are proprietary solutions. The main goal of our work is to create three tools, namely collector, export and mediator, which would have open code that could be used for many other experiments with monitoring the IoT environment.

The implementation of the exporter was realized for three IoT platforms, namely Arduino, Raspberry Pi and LoRa gateway, which was used for experimental verification of the possibility of data transfer of the TinyIPFIX protocol from IoT devices. However, the main benefit of this work is the implementation of a universal collector, which also includes the functionality of a mediator that provide mediation between TinyIPFIX and IPFIX data format. Experiments to verify the correctness of the implementation were performed based on the following architecture shown in Fig. 3.

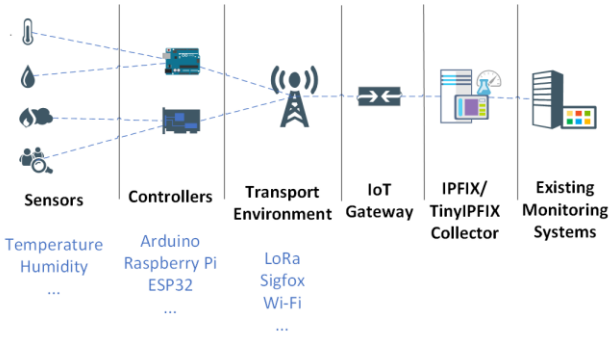

Fig. 3 Architecture for experiments

The experiment scenario consists of the following steps.

1. Start the export process on IoT devices (Arduino, Raspberry Pi, LoRa device), starting the universal collector and mediator on the LoRa gateway, starting the IPFixCol2 collector on a Linux device.
2. Create a file containing data about data flows stored in JSON format and save it to the IoT device.
3. Link the export process to the source file created in the previous step. Sending read data in TinyIPFIX format to the universal collector.
4. Receiving and storing data on a universal collector.
5. On the universal colector connect the collection process with the mediation process and sending of data in IPFIX format to an IPFIX collector IPFIXCol2 located in a common network.
6. Data storage on the IPFIXCol2 collector.
7. Comparison of the original data created in step two with the data stored on our universal collector and on the IPFIXCol2 collector.

The experiments confirmed the correctness of the implementation of the protocols by the fact that there was no change in the content of the data during the export process, collection on the universal collector or after the mediation process. The success of the implementation is also confirmed by the record from the Wireshark tool, which recognized the IPFIX data sent from the universal collector, as can be seen in Fig. 4.

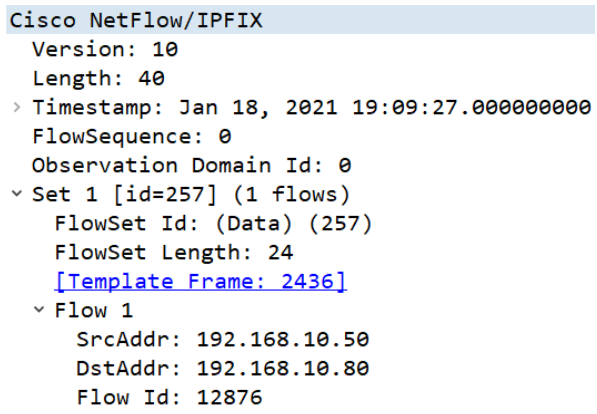

Fig. 4 Output from Wireshark tool

As can be seen, the successful identification of the protocol and the display of the transmitted data without an error message in the correct format confirms the correct implementation exactly according to the RFC 7011 standard. However, by using the organization number, it is possible to transfer any data, which was confirmed by other experiments.

Table 1 Comparasion of TinyIPFIX and IPFIX bandwidth conspumtion

	IPFIX	**TinyIPFIX**
Header	128b (16B)	Min: 24b (3B) Max: 40b (5B)
Set Header	32b (4B)	16b (2B)
Template Header	32b (4B)	15b
Specificator Field	*64b (8B)*	*64b (8B)*

Template		
Total	192b (24B)	Min: 56b (7B) Max: 72b (9B)
Saving		Min: 120b (15B) Max: 136b (17B)

Data		
Total	160b (20B)	Min: 40b (5B) Max: 56b (7B)
Saving		Min: 104b (13B) Max: 120b (15B)

As can be seen from the table above, using the TinyIPFIX protocol can save up to 17B when transmitting a template that defines the meaning of data and 15B when transmitting the content of transmitted data, which represents a significant saving of bandwidth consumption for IoT devices. This fact clearly confirms the suitability of the TinyIPFIX protocol for use in data transport in the IoT environment.

4. CONCLUSIONS

Based on the performed experiments, it can be argued that the implementation of the tools of the exporter, collector, and mediator to support the TinyIPFIX protocol for the IoT environment was successful. This demonstrates the successful data transfer without modification from the device in the IoT environment to the monitoring device located in a normal network environment. The experiments also confirmed a significant saving of bandwidth consumption, where the TinyIPFIX protocol can save bandwidth in the range of 63 - 72%. Thanks to the implementation of the TinyIPFIX protocol for common Linux systems, it is possible to continue research by carrying out several other experiments that could contribute to the creation of effective monitoring tools for common network environments, where they could significantly save bandwidth and thus improve conditions for quality of service.

ACKNOWLEDGMENTS

This publication was realized with support of the Operational Programme Integrated Infrastructure in frame of the project: Intelligent systems for UAV real-time operation and data processing, code ITMS2014+: 313011V422 and co-financed by the Europen Regional Development Found.

REFERENCES

[1] LI, J. – ZHANG, Y. – CHEN, X. – XIANG, Y.: *Secure attribute based data sharing for resource-limited users in cloud computing*, Computers & Security 72, No. (2018) 1–12 doi:10.1016/j.cose.2017.08.007.

[2] SHEN, J. – GUI, Z. – JI, S. – SHEN, J. – TAN, H. – TANG, Y.: *Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks*, Journal of Network and Computer Applications 106, No. (2018) 117–123 doi:10.1016/j.jnca.2018.01.003.

[3] AITKEN, P. – CLAISE, B. – TRAMMELL, B.: *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information* [RFC 7011], RFC Editor, No. 7011 (2013) doi:10.17487/RFC7011.

[4] SCHMITT, C. – STILLER, B. – TRAMMELL, B.: *TinyIPFIX for Smart Meters in Constrained Networks* [RFC 8272], RFC Editor, No. 8272 (2017) doi:10.17487/RFC8272.

[5] PETIJA, R. – GLEVANAK, M. – KUCAN, M. – FECILAK, P. – JAKAB, F.: *Experimental Implementation of TinyIPFIX Protocol for Arduino and Raspberry Pi Platform*, 8th International Conference on Emerging eLearning Technologies and Applications (ICETA), No. (2020) doi:10.1109/iceta51985.2020.9379188.

[6] PETIJA, R. – FECIĽAK, P. – JAKAB, F. – MICHALKO, M.: Critical analysis of Communication Protocols to Support the Quality of Services in IoT-based Infrastructures, 17th International Conference on Emerging eLearning Technologies and Applications (ICETA), No. (2019) doi:10.1109/iceta48886.2019.9039989.

[7] SCHMITT, C. – KOTHMAYR, T. – ERTL, B. – HU, W. – BRAUN, L. – CARLE, G.: TinyIPFIX: An efficient application protocol for data exchange in cyber physical systems, Computer Communications 74, No. (2016) 63-76 doi:10.1016/j.comcom.2014.05.012.

Received August 5, 2021, accepted September 27, 2021

BIOGRAPHIES

Rastislav Petija is PhD. student at the Technical University of Kosice, Slovakia. He earned a Master of Computer Science degree in 2018 at the Department of Computers and Informatics at the Technical University of Kosice. Currently, he is continuing his Ph.D. research in communication architecture in the IoT environment at the Department of Computers and Informatics, the Technical University of Kosice. He has published fifteen scientific papers. He is also Instructor trainer in several networking courses including DevNet Associate, CyberOps Associate, Network Security, CCNAv7 and CCNP Enterprise.

František Jakab is Director of the University Science Park TECHNICOM at the Technical University of Kosice, Slovakia. He has graduated from the Faculty of Computer Science and Electrical Engineering at the St. Petersburg Institute of the Electrical Engineering in the Field of System Engineering (Russian Federation). Main areas of his research activities: computer networks, new form of multimediabased communication. He is renowned author of more than 200 scientific publications and textbooks.

Peter Feciľak is Educator and researcher at Technical University of Kosice, Slovakia. In 2006 graduated (MSc.) at Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Kosice. In 2009, he finished his PhD studies at the same department with the focus on optimization of computer networks. Currently, he is working as associate professor at DCI, FEI, Technical University of Kosice. His current teaching and research interests are computer networks, network monitoring, quality of services, IoT and smart grid systems.

Miroslav Michalko is Head of Computer Networks Laboratory at Department of Computers and Informatics at Technical University of Kosice, Slovakia. He graduated master and PhD. in field of Informatics at the same university. Main areas of his research activities are computer networks, video streaming services and smart solutions for cities using Internet of Things. He is project manager of various national and international projects and published almost 100 scientific publications.