Method for OpenFlow Protocol Verification

Olena Tkachova, Isaam Saad
Telecommunication system department
Kharkov National University of Radio Electronics
Kharkov, Ukraine
korov4enko@mail.ru, 71dkh@mail.ru

***Abstract* — This paper is devoted to SDNs verification. The main problems in deployment and maintenance process are analyzed. Methods of formal verification are suggest as way to check correctness and corresponds to the requirement. The modification of Model Checking proposed for verification the main feature of OpenFlow protocol.**

Keywords—OpenFlow; Model Checking; virafication; simbolic execution; Software-Defined Networking

I. INTRODUCTION

Services providing in modern communication networks is a complex process. Many different protocols involved in the process of data transfer. The protocols operate inside the different types of network equipment: routers, switches, servers, firewalls etc. Each developer of network equipment make his own adjustments. As a result many problems can occur during operation process. The problems connect with arising equipment complex, correct network operation and and service availability [1]. The concept of Software-Defined Networking (SDN) is designed to simplify the process of sending data. SDN concept proposes to separate control and management function (control plan) from data transmission function (data plan). SDN uses a logically-centralized programming element or controller and open standard protocols. However, at the same time, many errors remain unresolved.

Analysis of existing solutions [2-3] showed that the majority of errors occur due to differences in the protocols functionality and realisation of operation algorithms, different interpretations of specifications, compatibility requirements and logic errors in the design of networks.

Verification is one of the methods of errors detection. The main purpose of verification is to prove that the result of operation of the network is almost compliant with the tasks defined in the specification [4]. Many different types of verification are using today. The basic verification methods are testing or executing different scenarios and formal methods or formal verification. The disadvantage of testing is inability to verify all boundary services states.

Formal approach in the verification process allows systematical search and analysis of the set of states that can be reached during the operation of the network model or protocols. The advantage of such formal methods is their expressive power. Any system of interacting processes can be formed using formal methods. The mathematical logic is foundation of formal methods. Unfortunately, the application of basic logics does not allow full priority to formalize the processes during their operation, availability of resources for SDN verification.

Thus, it is necessary to develop a method of formalization and verification of SDN, that allows formalize complex characteristics and properties of SDN.

II. METHOD OF OPENFLOW PROTOCOL VERIFICATION

The main different SDNs from traditional network is separation control plan from data plan [1]. OpenFlow switch is the middle network equipment that forward data between control and data plan. OpenFlow is an open protocol that responsible for the transmission of control information and the formation of forwarding path. OpenFlow protocol has different versions. Network faults can occurs when sharing various versions of the protocol. Therefore, much attention should be paid to OpenFlow protocol verification.

The main aim of formal verification is the proof that the result of service or protocol implementation corresponds to the list of requirements that defined in his specification [5]:

$$V: M_S \equiv M_R <=> (\{m_{Si}\},\{m_{Ri}\}), \quad (1)$$

where M_R - model of OpenFlow protocol implementation, m_{Ri} - i-th state of protocol or service implementation, M_S - model of specification, m_{Si} - i-th state of specification requirements.

In general, the verification steps can be represented as diagrams that shown on Figure 1.

Input data are the requirements of the specification. Input data typically represented as a subset of natural language or as UML-, SDL- diagrams. This possible ambiguity in the interpretation of requirements by different developers that initially leads to errors.

In the process of verification of compliance checks implementation of the model protocol or service specification requirements.

In the case when mismatch is detected counterexample will be construct (reveals behavior that leads to an incorrect sequence).

Functional of SDN services and protocols is constantly updated. All time new option is added. Thus, the following sub steps may be formed through conducting verification:

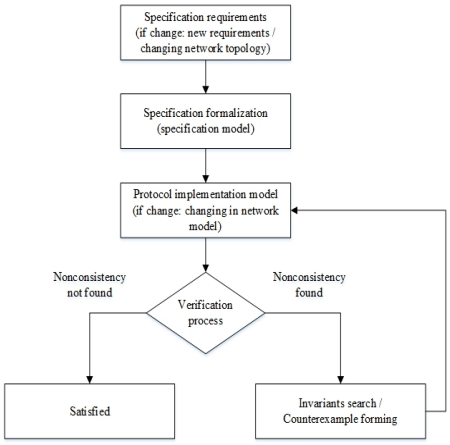

Fig. 1. Algorithm of verification process

1. verification the new service or protocol developed for compliance the requirements of the specification;

SDNs have a dynamic nature. Therefore set of protocol elements S_{SDN}, corresponds to a set of individual processes P_{SDN}. The logical systems of elements interaction in time (f, S_{SDN}, P_{SDN}) allows to formalize and analyze correctness and compliance step by step.

This corresponds to the desired properties is guaranteed when the requirement is met:

$$(f, S_{SDN}, P_{SDN})_{mod} \equiv (f, S_{SDN}, P_{SDN})_{spec}$$

where $(f, S_{SDN}, P_{SDN})_{mod}$ is a set that define the relationship of states between OpenFlow protocol elements and their sequence change for OpenFlow model, $(f, S_{SDN}, P_{SDN})_{spec}$ is a set that define the relationship of states OpenFlow protocol elements and their sequence change for specification.

2. verification of different specifications versions of OpenFlow protocol (v.1.1, v.1.3) for compliance and consistency requirements.

Formal verification of specifications of different versions can be represented as follows:

$$(f, S_{SDN}, P_{SDN})'_{spec} \equiv (f, S_{SDN}, P_{SDN})''_{spec},$$

where $(f, S_{SDN}, P_{SDN})'_{spec}$ is a set of OpenFlow protocol interaction that correspond to specification version 1.1.0, for example, $(f, S_{SDN}, P_{SDN})''_{spec}$, $'_{spec}$ is a set of OpenFlow protocol interaction that correspond to specification version 1.3.0.

3. verification of conformity and consistency requirements for upgrade or implement a new service.

In this case, not all system elements take place in verification implementation - only updates the requirements. As a rule, such requirements affect only on local state of models. Formal verification in such case can be written as:

$$(fp, S_{SDNp}, P_{SDNp})_{prototype} \equiv \{(f, S_{SDN}, P_{SDN})_{mod}\},$$

where $(fp, S_{SDNp}, P_{SDNp})_{prototype}$ is a set of new requirements that should be implemented in OpenFlow protocol or all SDN network, $\{(f, S_{SDN}, P_{SDN})_{mod}\}$ – interaction sets from verified model.

One of the more appropriate approach for SDN verification is Model Checking [5]. Algorithms that systematically explore all states of the system model accompany the specification and protocol implementation models. This provides the basis for a whole range of verification techniques ranging from an exhaustive exploration (model checking) to experiments with a restrictive set of scenarios in the model (simulation) and also allows to analyze all model properties (dead lock, liveness, reachable state).

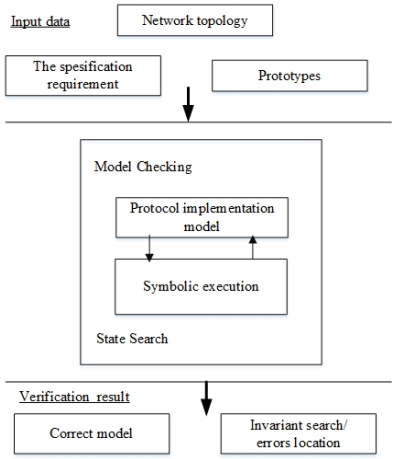

Fig. 2. Modification of Model Checking approach

According to the SDN nature proposed, construct prototypes. Prototypes are the rules that fixed (formally determined) set of requirements for an individual property protocol. Prototypes using allows to reduce the set of verifiable conditions.

Proposed approach allows taking into account visualization. Each virtual topology can be modeled step-by-step during the verification process. In this way, the NP-complete task can be avoided.

REFERENCES

[1] ITMO OpenFlow tests repository. [Online]. Available: https://github.com/itmo-infocom/of-tests

[2] David L. Stevens. Preparing a customer's network for iSCSI: Five points to consider. [Online]. Available: http://searchitchannel.techtarget.com/tip/Preparing-acustomers-network-for-iSCSI-Five-points-to-consider

[3] Google. Inter-Datacenter WAN with centralized TE using SDN and OpenFlow. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdnresources/customer-case-studies/cs-googlesdn.pdf

[4] OpenFlow specifications. [Online]. Available: https://www.opennetworking.org/sdn-resources/onfspecifications/Openflow

[5] R. Alur and R. K. Brayton and T. Henzinger and S. Qadeer and S. K. Rajamani. Partial order reduction in symbolic state-space exploration. Formal Methods in System Design, 18(2):97–116, 2001.