ETSCP: Flexible SDN data plane configuration based on bootstrapping of in-band control channels

Romerson Deiny Oliveira
Department of Computer Science
Unimontes
Montes Claros, Brazil
romerson.oliveira@unimontes.br

Marcelo Silva Freitas
Department of Exact Sciences
Federal University of Jataí (UFJ)
Jataí, Brazil
msfreitas@ufg.br

Diego Nunes Molinos
Faculty of Computing
Federal University of Uberlândia (UFU)
Uberlândia, Brazil
diego.molinos@ufu.br

Pedro Frosi Rosa
Faculty of Computing
Federal University of Uberlândia (UFU)
Uberlândia, Brazil
pfrosi@ufu.br

Daniel Gomes Mesquita
Campus Santana do Livramento
Federal University of Pampa (Unipampa)
Santana do Livramento, Brazil
mesquita@unipampa.edu.br

***Abstract*—Decoupling control and data planes, as proposed by SDN environments, enables the improvement of network programmability not only at the control plane but also in the forwarding mechanisms. In this sense, hardware designs and new protocols can increase the flexibility offered by the whole infrastructure. Despite of many benefits provided by SDN architectures, there still are some points that can be further addressed, such as flexibility and programmability at the data plane level and networking device bootstrapping. Moved by these issues, this paper presents a proposal for a protocol, named ETSCP, to program the elements of a flexible data plane, supported by autonomous initialization of control plane. The elements of ETSCP are presented, as well as a simulation and a validation system built to demonstrate its operation in a local network.**

***Index Terms*—SDN, ETSCP, bootstrapping, in-band control plane, ETArch.**

I. Introduction

Due to the decoupling of the data plane and control plane in Software-Defined Networks (SDN) a natural distance from the functions implemented in hardware and software is created. It is making the hardware's behavior to be a single granular network fabric and all the primary functions of the network are directed to the software level. This scenario has contributed so that all flexibility and programmability were carried over to the network control level, being practically unexplored at the data plane level, automatically neglecting its capacity in terms of programmability and flexibility. The fact is that relying entirely on the software layer to provide flexibility and programmability can result in loss of quality of service (QoS).

In this context, the Entity Title Architecture (ETArch) aims to offer an architecture with features and support to context-oriented algorithms that allow changes and adjustments according to the network requirements [1] [2]. It was intended to deal with Future Internet requirements such as multicasting, mobility and QoS [2] [1].

978-3-903176-32-4 © 2021 IFIP

Improvements related to the control plane and the management plane of ETArch have been devised. An important step towards that improvements was the creation of the ConForm protocol [3], a new protocol to autonomously discover the network topology and configure in-band control channels from the controller to devices.

Another fundamental advance for the realization of the ETArch architecture, this in the data plane, motivated by the limitations of programmability of current SDN switches, was the design of an ETArch switch and a protocol that would allow the controller to configure it, namely, the ETArch Switch Configuration Protocol (ETSCP).

The goal of this work was to implement and validate ETSCP protocol. The validation was conducted by a OMNeT++ [4] simulation of the network bootstrapping, by ConForm protocol, followed by switch configuration tests using ETSCP protocol to illustrate the correct functioning as a whole. Additionally , a real proof of concept was done by a FPGA implementation of ETSCP running in a local ETArch network.

II. Background

This section provides a desirable background about the network environment in with ETSCP operates.

A. Domain Title Service Environment

Entity Title Architecture(ETArch) has been designed in order to approximate semantically the Application layer to the lower layers. ETArch is based on Software Defined Networks (SDN), and presents innovations related to addressing of network elements by the separation of location and identification, which intrinsically enables multicast transmissions and network mobility [2].

From the ETArch perspective, it's important to comprehend the concepts of: *i)* ***Entity***: defined as any element that

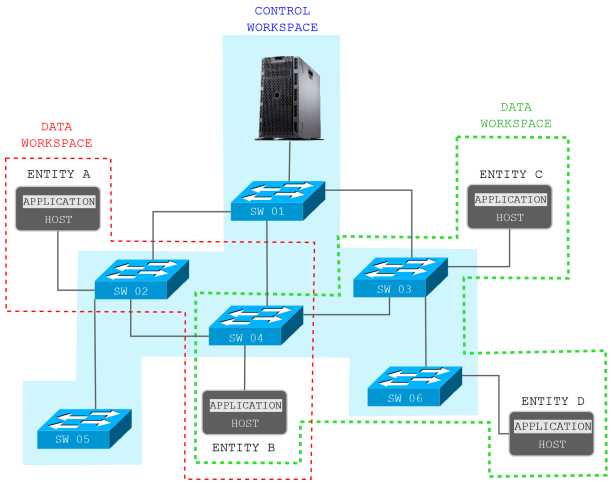

Fig. 1: ETArch typical environment.

wants to communicate in a distributed environment such as applications, devices, computers, a network element or even a sensor [1]; *ii)* ***Workspaces***: logical multi-end communication buses that are independent of the network topology. Entities that wish to communicate are added in a specific Workspace [2]. They are driven by the application and carries a set of capabilities of the communication domain; and *iii)* ***Domain Title Service - DTS***: A DTS represents a set of ETArch controllers in a distributed system. The goal of DTS agents is separate the network into manageable subnets within ETArch [2]. A Domain Title Service Agent (DTSA) acts in the control plane, managing the network elements. DTSA is responsible for creating, managing and dropping Workspaces on network elements.

B. Control Plane and Network Formation Protocol

The ConForm (Control Plane and Network Formation) protocol was motivated by the lack of a standard initialization procedure of the SDN control plane, independent of other protocols, able to autonomously configure in-band control channels from controller to network devices.

In summary, the services of the ConForm protocol enables a network device to register itself with the network controller simultaneously establishing an in-band control channel with it. Then, the device start to discover its neighbors, updating its internal table. Once this local discovery complete, the contents of the device table is delivered to the controller. After that, the device will be in ACTIVE state and able to assist on the activation of neighboring devices. Thus, an activation wave propagates through the network infrastructure making all devices registered, activated and operational. In addition, the information collected by the controller, sent by the devices, is used to compose the logical representation of the network topology. Full documentation about ConForm protocol can be found at [3].

TABLE I: ETSCP Services and Vocabulary

Service	Messages	Functional Description
Create Workspace	*AddWkpREQ* *AddWkpRESP*	Used by DTSAs to require the creation of new Workspaces in each switch.
Edit Workspace	*EditWkpREQ* *EditWkpRESP*	Used by DTSAs to modify parameters of the Workspace.
Remove Workspace	*RemWkpREQ* *RemWkpRESP*	Used by DTSAs to remove a Workspace. and its entry from a Workspace table.
Change Frequency	*EditFreqREQ* *EditFreqRESP*	Used by DTSA to modify the frequency of the switch's schedulers.

III. ETArch Switch Configuration Protocol

Building a network standard communication means that we must reach a consistent and well formed protocol. This is achieved when five elements are considered in the protocol specification [5]. In this sense, the five elements of the ETArch Switch Configuration Protocol (ETSCP) are presented in this section: assumptions about environment, services, vocabulary, formatting and procedure rules.

A. Assumptions about the Environment

ETSCP was designed to manage the communication between a SDN controller and a switch in an DTS domain. Hereby, an ETArch setup is a typical environment for ETSCP operation and Fig. 1 provides an overview of this domain.

Each ETSCP instance running in a network switch is stand alone, i.e., no matter how many switches the infrastructure has, each instance runs independently as an atom in a distributed system. On the two sides of the protocol communication there are the DTSA and the switch. There is no application entity involved in this communication, it means that no users applications uses ETSCP. This is intuitive in certain way, since user entities do not change configurations of the switch, only DTSAs are supposed to do that.

Furthermore, it is a peer to peer protocol and this means that if a DTSA needs to change configurations in several switches simultaneously, this DTSA must call the protocol ETSCP several times and send messages to each switch separately, each message with different field parameters. Messages reach the switch as frames and are processed by the control unit.

B. Services and Vocabulary

The ETSCP design was focused on having a simple, well-formed, robust and consistent protocol, but with a minimum number of messages to minimize the impact on the switch design and controller overhead. Table I gives an accurate relation between the protocol services and their respective messages. All the services are confirmed and the messages follow the taxonomy of confirmed services: `Request/Indication` and `Response/Confirmation` have the same content differing semantically from the moment it occurs.

C. Format

The essence of ETSCP is to configure a switch. It means that the messages are to resolve a configuration protocol. In

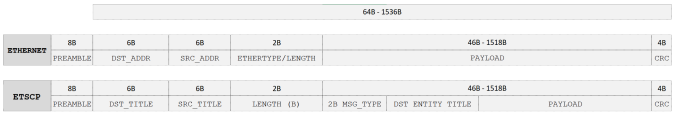

Fig. 2: ETSCP message fields.

fact, these messages are frames and do not overpass the link level. Message format is depicted in Fig. 2. It is remarkable to explain by this point that the ETSCP implements the horizontal addressing with title, but here the format is similar to IEEE Std 802.3 [6] to keep up compatibility with the current network interface cards, allowing the reuse of available technology and preserving investments. As we are editing the lowest level of abstraction, i.e., the link level, where hardware and software meet each other in the network implementations, the frame could be formatted in a more flexible and completely different way.

As stated before, messages fundamentally carries control information. That is why the header and payload have similar lengths. Sometimes, more header than payload. The fields are depicted as follow: *i)* `DEST_TITLE`: Workspace destination title; *ii)* `SRC_TITLE`: Entity source title; *iii)* `LENGTH`: Length, in bytes, of the payload; *iv)* `MSG_TYPE`: Type of message, according to Table I; *v)* `DEST_ENTITY_TITLE`: Title of the destination entity, joining a Workspace, that must read the content of the message; *vi)* `PAYLOAD`: Data field to message parameters (such as Workspace title, portmap, bitmap, etc.); and *vii)* `CRC`: Error verification code. There is no error correction at this version, but detection.

D. Procedure Rules

In this section, the services from Table I are better explained. The protocol rules are specified through a Finite State Machine (FSM). The diagram in Fig. 3 formally describe the behavior of the protocol services. Each transition is represented by the combination of input and output in the format $\frac{input}{output}$. The symbols `?` and `!` represent, respectively, the actions of *Receiving* and *Sending* messages.

The state `ACTIVE` is the initial and final state, i.e., the default behavior. If no message is received, the switch never leaves this state, as well as when all messages are exchanged, the final transition leads to `ACTIVE`. All the other states are depicted in different colors according to each service execution.

When a switch is active, it means that all the setting up process is completed by ConForm and, from this point, it can respond to DTSA requests and configure Workspaces. No manual intervention is required at this stage. Prior to it, the well known Workspace names must have been configured.

1) Create Workspace: It can be activated both when the DTSA needs to create a control logic domain or when responding to an entity request to create a data Workspace. Application entities cannot create Workspaces, they must ask directly to the DTSA, then DTSA triggers ETSCP's service `Create Workspace`.

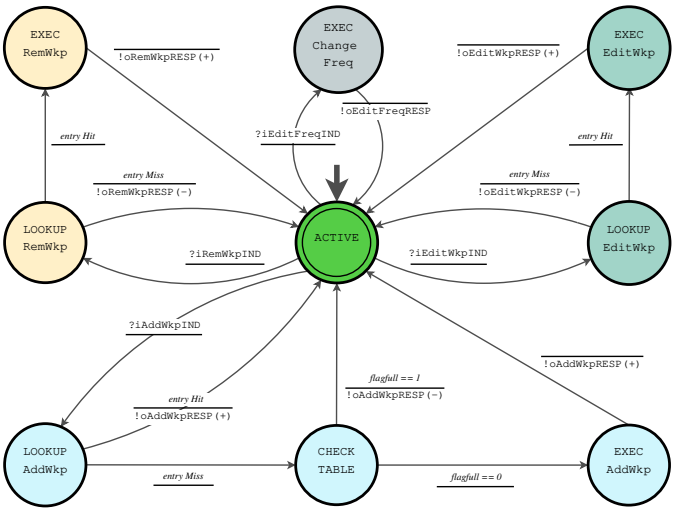

Fig. 3: ETSCP Finite State Machine.

At first, DTSA sends a `AddWkp_Req` that reaches the switch as `AddWkp_Ind`. When it occurs, the switch goes to the state `Lookup AddWkp` to check if the Workspace request is already a entry in the table. In the case of duplicated requisition, and a entry hit occurs, the switch itself sends a `AddWKP_Resp(+)`, meaning that the Workspace is already created. In case of no entry found, then the switch goes to the state `Check Table` to verify if is it possible to add a new entry. If not, a negative response `AddWkp_Resp(-)` is sent to DTSA. Otherwise, the Workspace is created and a positive response `AddWkp_Resp(+)` is sent.

2) Edit Workspace: At first, DTSA sends a message `EditWkp_Req` to the switch. Then, if the referred Workspace is found in the table, the edition is completed and a positive response `EditWKP_Resp(+)` is sent back to DTSA. Otherwise, a negative response is sent.

3) Remove Workspace: If a Workspace is no longer necessary, the DTSA can remove it from the switch tables by sending a request `RemWkp_Req`. When received the `RemWkp_Ind`, the switch searches for this entry in the table and removes it, then a positive response `RemWkp_Resp(+)` is sent back to DTSA. Otherwise, a negative response `RemWkp_Resp(-)` is sent instead.

4) Change Frequency: Related to the switch architecture and organization. The switch's forwarding plane is made of more then one scheduler. This is one of the resources that enables multiple virtual data planes to act as Workspaces [7]. When DTSA wants to benefit one Workspace (or a class of them) in terms of bandwidth, it is possible to change the frequency the scheduler moves forward frames to this particular Workspace. By sending a message `EditFreq_Req`, the DTSA means that the frequency divider must be reconfigured to a faster or slower rate than the current one. If a `EditFreq_Ind` is received, the switch automatically proceed with the hardware change and sends back a `EditFreq_Resp`. An independent paper will bring further details about the switch architecture.

IV. Simulation and Prototyping Experimentation

This research combine two protocols addressing different moments of the network operation. ConForm, the SDN bootstrapping control protocol, specify how to configure a control plane within a regular (in-band) SDN infrastructure by building an ideal spanning tree as a logical multicast domain, i.e., a Control Workspace. ETSCP, the switch configuration protocol, acts at the following moment, to manage creation and maintenance of other logic domains, named Data Workspaces, inside the switches.

Two methods are used in this paper to provide a whole demonstration. At first, focusing on the network setup, a simulation built with OMNeT++ illustrates how the bootstrapping goes and interacts with ETSCP. Secondly, ETSCP is used to configure a FPGA prototyped switch.

A. ConForm with ETSCP Running on OMNeT++

The model implemented on the OMNeT++ platform consists of two classes, `Controller` and `Switch`, which implement the FSM of the ConForm and ETSCP protocols. Such classes are used to instantiate some topology to simulation of a network with one controller and a few switches. A video of the simulation was made available [8].

In short, the video shows the simulation of the bootstrapping conducted by the ConForm protocol, configuring the in-band control channels, that is, creating a control Workspace in the network domain, and then shows the simulation of the establishment of a data Workspace including two entities $E1$ and $E2$, connected to different switches. More details can be found in the video description.

B. FPGA-based Validation System

A local network was built to obtain a proof of concept of ETSCP and create, maintain and change parameters of Workspaces. At this time, it was not critical to pursue strict performance parameters, although they are guiding the implementation. Fig. 4 depicts the system.

There is an ETArch switch in the center. It is a first version of a switch with a new approach in the Ethernet sub-layer of Media Access Control, the lowest level before the physical layer. It reflects the design of a switch Ethernet and TCP/IP-independent, versatile for establishing distinct Workspaces with different hardware-level features [7].

ETSCP is implemented in VHDL and there is no software processing for message parsing. The goal is to offer a switch that allows different methods of medium access control and differentiate diverse communication needs, allowing the parameterization of each Workspace. Control and forwarding planes remains separated, but the programmability is also added to the forwarding plane with the possibility of change hardware configurations while running.

It is prototyped in reconfigurable hardware, using an Altera Cyclone II FPGA. To bypass MAC and Ethernet controllers, switch ports were connected with external RJ45 transceivers completely raw of protocols. Hence, all the hardware manipulation is written in ETArch specifications.

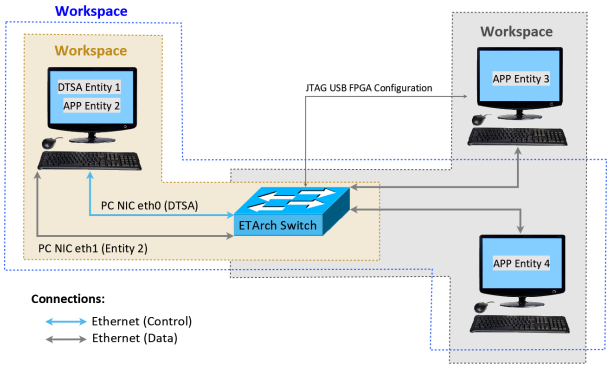

Fig. 4: Validation System to ETSCP.

Also, according to Fig. 4, three personal computers were connected to the switch as entities. For the DTSA, a lightweight version is emulated with the ETSCP message handler in operation. The PCs are running either Ubuntu 18.04 or Windows 7. DTSA uses Ostinato [9] to inject messages. Wireshark is running in all of them to capture message exchange.

Before any message exchanging, only the well known control Workspaces are set as table entries. Also, the switch already knows DTSA title, automatically settled up during bootstrap with ConForm. No data Workspace is set at the beginning. All the messages according to Table I were noticed in the DE2 displays and leds, aw well as in the Wireshark. Up to this moment, no stress tests were made. These results presented here stand as the proof of concept and leads us to migrate to a faster hardware platform.

V. Concluding Remarks

This papers presents the ETArch Switch Configuration Protocol to create and modify logic channels in the data plane of a SDN network. The ETSCP guides a switch that brings a medium access control different from Ethernet and supports Workspace's management in operation time.

The control channels configured by ConForm protocol during network bootstrapping are the logical ways in which ETSCP protocol messages travel to reach the devices being controlled without depending on another protocol to do so. The network topology view, also enabled by ConForm) allows important ETArch services to be implemented in an optimized way, such as, for example, the discovery of the shortest path to extend an existing Data Workspace to a new entity that wants to be attached to it.

To the road ahead, we look forward to use a high performance FPGA as the switch engine and have the ETSCP implementation co-designed in an embedded software environment.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 and built with the support of MEHAR team.

REFERENCES

[1] F. de Oliveira Silva, M. A. Gonçalves, J. H. de Souza Pereira, R. Pasquini, P. F. Rosa, and S. T. Kofuji, "On the analysis of multicast traffic over the entity title architecture," in *2012 18th IEEE International Conference on Networks (ICON)*, Dec 2012, pp. 30–35.

[2] F. de Oliveira Silva, J. H. d. S. Pereira, P. F. Rosa, and S. T. Kofuji, "Enabling future internet architecture research and experimentation by using software defined networking," in *2012 European Workshop on Software Defined Networking*, Oct 2012, pp. 73–78.

[3] M. S. Freitas, P. F. Rosa, and F. de Oliveira Silva, "ConForm: In-Band Control Flows Self-establishment with Integrated Topology Discovery to SDN-Based Networks," in *Web, Artificial Intelligence and Network Applications*, L. Barolli, F. Amato, F. Moscato, T. Enokido, and M. Takizawa, Eds. Cham: Springer International Publishing, 2020, vol. 1150, pp. 100–109, series Title: Advances in Intelligent Systems and Computing. [Online]. Available: http://link.springer.com/10.1007/978-3-030-44038-1_10

[4] A. Virdis and M. Kirsche, *Recent advances in network simulation: the OMNeT++ environment and its ecosystem*. Springer International Publishing, 2019, oCLC: 1082225057.

[5] G. J. Holzmann, *Design and validation of computer protocols*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1991.

[6] IEEE, "Ieee standard for ethernet," *IEEE Std 802.3-2018 (Revision of IEEE Std 802.3-2015)*, pp. 1–5600, Aug 2018.

[7] R. D. Oliveira., D. N. Molinos., M. S. Freitas., P. F. Rosa., and F. de Oliveira Silva., "Workspace-based virtual networks: A clean slate approach to slicing cloud networks," in *Proceedings of the 9th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER,*, INSTICC. SciTePress, 2019, pp. 464–470. [Online]. Available: https://doi.org/10.5220/0007753104640470

[8] R. D. Oliveira., D. N. Molinos., M. S. Freitas., P. F. Rosa., and D. G. Mesquita., "Simulation of conform-etscp operation," URL https://youtu.be/cViUo5ExuUs, Jan. 2021.

[9] P. Srivats, "Ostinato packet generator," URL: https://ostinato.org/, 2019.