

Specialized Network Self-configuration: An Approach using Self-Organizing Networks Architecture (SONAr)

Daniel Ricardo Cunha Oliveira^a, Maurício Amaral Gonçalves^b, Natal Vieira de Souza Neto^c, Flávio de Oliveira Silva^d and Pedro Frosi Rosa^e

Faculty of Computing, Federal University of Uberlândia, Uberlândia, Brazil

Keywords: Self-management, Self-configuration, Self-Organizing Networks, SDN, NFV, Cloud Computing.

Abstract: The increasing complexity of computer networks is constantly requiring the proposal of new network automation models. These are mostly based on Software-Defined Networks (SDN) and Network Functions Virtualization (NFV) with implementation of the features called self-* within the concept of Self-Organizing Networks (SON), which includes self-configuration. Regarding this matter, some models have already been proposed at a higher level of definition, but there is still a lot of practical work to be done. In this paper, we present the modeling, implementation and performance of a network self-configuration system when a new edge element, which demands a specialized type of configuration, has been inserted. This is a typical case of a telecom network, in which edge network elements (NEs) usually require a very specific set of network parameters to be configured. The system recognizes the element and implements this specific network segment parameterization, acting as a specialized "plug-and-play" feature. The approach will be based on the Self-Organized Networks Architecture (SONAr) framework, which was very suitable for this functionality.

1 INTRODUCTION

If there is an unquestionable fact in the recent history of technology, it is the fast growth of applications and the densification of devices connected to the Internet, especially if we take into account access on mobile devices. This growth tends to remain at exponential levels, especially with technological adventures such as 8K ultra-resolution TV, new virtual/augmented reality devices, cloud gaming, massive Internet of Things (IoT) and other applications that new access technologies, e.g. 5G, will support. All this evolution brings intrinsic concerns with the infrastructure that will support all this load.

In 2001, IBM demonstrated an apprehension with the increase in the complexity of computer networks and the respective increase in cost that it implies, in an article that defines the bases and concepts for autonomic computing, including the concepts of self-* (Ganek and Corbi, 2003), namely self-

configuration, self-healing, self-protection and self-optimization, which defines the basis of "autonomic computing".

In this work, we aim at a Self-Organizing Networks (SON) implementation focused in the first property, the self-configuration, which was implemented in the context of a more complete framework called Self-Organizing Networks Architecture (SONAr), which provides the basis for implementing the self-* functionalities that make up a SON system.

Furthermore, this proposal for a self-configuration system will focus on network elements that require a highly specialized network configuration to support them (rather than the more usual generic network elements), typically edge elements in a telecommunications network. It will present a model for describing this specialized configuration and the automated process to implement it on a network without human intervention.

2 BACKGROUND

In general, the term SON was popularized in the context of automation of mobile network management,

^a  <https://orcid.org/0000-0003-4767-5518>

^b  <https://orcid.org/0000-0002-6985-638X>

^c  <https://orcid.org/0000-0001-5047-4106>

^d  <https://orcid.org/0000-0001-7051-7396>

^e  <https://orcid.org/0000-0001-8820-9113>

having been standardized by 3GPP as a solution for the administration of the increasing quantity and complexity of network parameters since Release 8 (3GPP, 2018b) (3GPP, 2018a).

The SON properties in mobile networks allow the minimization of human intervention in certain aspects of its management. However, SON functionalities in mobile networks concern the parameterization of access network configurations and not extend to network core elements (TELUS and Huawei, 2016).

2.1 SDN and NFV Paradigm

SDN and NFV concepts were born out of the industry requirements to evolve traditional network architectures into more flexible models that would allow a greater degree of automation and freedom from vendors. The first allows the infrastructure to be abstracted into network applications and services, so that it can be treated as a logical or virtual entity (Open Networking Foundation, 2012) and the later involves the implementation of network functions in software that can run on a range of industry standard server hardware (Chiosi et al., 2012).

In this work, it is assumed the adherence of both paradigms in the target networks.

2.2 Related Work

An approach towards establishing a framework and a reference architecture for SON combining the SDN and NFV approaches was made in (Neves et al., 2016), with the SELFNET project. It proposes high level SON functions, focusing on self-healing, self-protection and self-optimization, with more emphasis on the latter, with self-configuration not being treated. The article itself does not present practical implementations or results, however other later publications already show measured results, especially dealing with self-optimization (Jiang et al., 2017).

Superfluidity is a project within Horizon 2020 program and aims to present a flexible and adaptive network architecture, focusing mainly on important aspects for 5G, in a context of NFV and SDN networks (Bianchi and *et al.*, 2016). It defines logical entities that perform a set of functionalities and the objective is to obtain a formal and agnostic description of how these entities should work. Nevertheless, the work does not present any practical implementation proposal other than the framework itself.

(Katiyar et al., 2015) proposes a self-configuration mechanism for a new SDN switch inserted in hybrid SDN and non-SDN networks. The system was implemented and tested in a simple network model, but no

response time or convergence metrics were presented, and it is also not inserted in a larger context of SON. Finally, it does not address the self-configuration of specialized NEs.

To the best of our knowledge, no work with a complete range of the self-* properties – specially with a particular focus on self-configuration for specialized elements – has been presented yet.

3 SELF-ORGANIZING NETWORKS ARCHITECTURE (SONAr)

A SON system must run on a well-defined architecture. A practical framework and implementation was proposed in our previous work (Gonçalves et al., 2020), the Self-Organizing Networks Architecture (SONAr), which was chosen to be the basis for the specialized self-configuration features presented in this work.

The SONAr framework is shown in the Figure 1. It consists of five layers that have specific functions and functionalities, which will be described below. In the next subsections, a brief description of the layers and components of this framework will be presented and all component descriptions will be based on (Gonçalves et al., 2020), unless otherwise mentioned.

3.1 Client Layer and SDN Network Layer

In the Client Layer are placed the edge elements that will be added to the network and that will require an automatic configuration in order to allow full operation within it, a process which is the object of study of this work (specialized self-configuration).

The SDN Network Layer can be logically divided into two: the lower one represents an infrastructure sublayer, composed of the network elements themselves, e.g. switches, routers and bridges, whether they are physical or virtual elements, while the upper sub-layer, centralizing the control plane, hold the primary “intelligence” of the network.

3.2 Interface Layer

This layer comprises the Collecting Entities (CoEs) – composed of several modules that together collect metrics, topology information, alarms, samples, logs and results during the network operation – and other components that will be presented below.

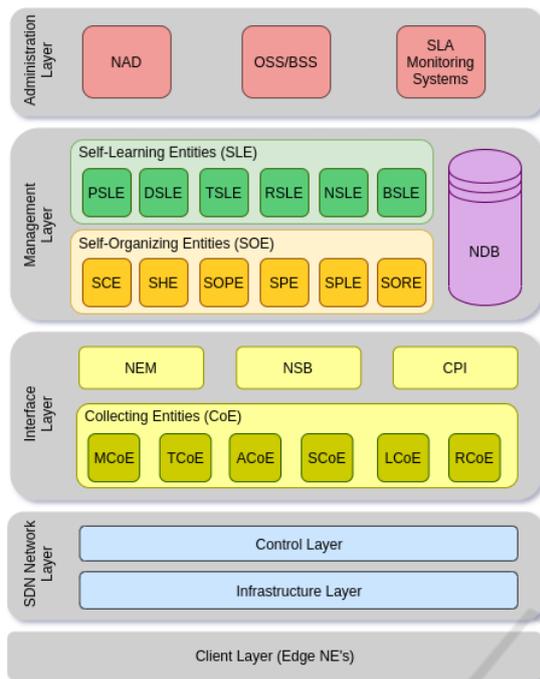


Figure 1: Simplified layered diagram of the SONAr architecture.

Network Event Manager (NEM): it is a publisher/subscriber provider that manages events, which, in turn, are categorized into topics. **Network Service Broker (NSB):** it is used to configure services based on intentions expressed in natural or formal language by customers or system administrators. **Control Primitives Interceptor (CPI):** It is located in the Southbound Interface (SBI), acting as a proxy between the network and the controller. It collects a large base of information related to network management and store it in the Network Database (NDB).

3.3 Management Layer

The management layer can be divided into two sub-layers: the first integrates the Self-Organizing Entities (SOE) and the last combines the Self-Learning Entities (SLE). SOEs are responsible for performing real-time algorithms related to the fundamentals of autonomous computing. The components of the SOE sub-layer are as follows.

Self-Configuration Entity (SCE) is the most relevant to this work. It acts in two moments, in the network bootstrapping process and when a new NE is plugged into it, in the process called plug-and-play. The last can be further divided into self-configuration of general cases and specific cases applicable to specialized edge NE (Oliveira et al., 2020), which will be the investigation topic of this work.

Self-Healing Entity (SHE) implements algorithms that allow to predict or detect failures in all the elements shown in the Figure 1, providing solutions for recovering them. **Self-Optimization Entity (SOPE)** aims to intervene in the network in order to maximize its performance, based on the collected and analyzed statistics. **Self-Protection Entity (SPE)** guarantees the protection of all layers presented in the Figure 1 against internal and external attacks. **Self-Planning Entity (SPLE)** organizes, controls and schedules all actions on the network coming from the other entities of this layer. **Self-Orchestration Entity (SORE)** has the role of receiving all actions from other entities and organizing them in queues, prioritizing these actions.

Above the SOE sub-layer is the SLE sub-layer. The purpose of these components is to provide AI-based mechanisms to overcome the absence of human intervention. Finally, the NDB implements the databases that aggregate all information generated and stored in SONAr.

3.4 Administration Layer

The Administration Layer makes up the last level of the SONAr. It implements the Network Administration Dashboard (NAD), where network operators can access SONAr data and administrate the system, Operations Support System (OSS), Business Support System (BSS) and Service Level Agreement (SLA) Monitoring Systems.

4 SPECIALIZED SELF-CONFIGURATION MODULE (SSCM)

The purpose of this work is the implementation of a module within the SCE in the SONAr system, i.e. Specialized Self-Configuration Module (SSCM).

The goal of SSCM is to allow for a specific configuration of a segment of a network in response to the insertion of a new network element that requires a specialized level of configuration, such as an edge element that requires specific network parameterization for its functioning. The module must identify this new NE (typically an edge element with telecom functions), check if there are specific configurations described for it and, if any, implement them on the network, acting as a specialized plug-and-play. It should be flexible for use in any practical cases, that is, support the configuration for any types of edge elements and all topology settings.

4.1 SSCM Domain and Architecture

The applicability of SSCM operation presupposes that there are two well-defined points for the beginning and ending of the specialized configuration: the first point is the exit interface of the edge network element itself (edge NE), and the second the exit interface of the element where the network converges, usually a gateway router or an aggregating switch, where the configuration process will stop. The final destination elements of the packets sent by the edge element can be inside or outside the SONAr domain and can be one or more, but the edge element itself must be unique (identifiable by its outgoing interface) and must also be necessarily within the SONAr domain. Both SONAr and SSCM domains are represented in the Figure 2.

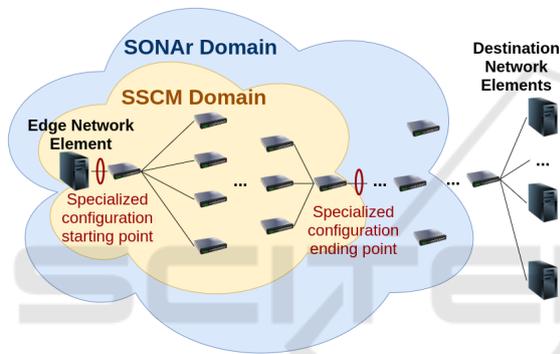


Figure 2: Topology requirements for SSCM applicability.

For the network to be configured correctly after the insertion of an edge element, a detailed description of this configuration must be informed to SONAr. This is done by an specific entry in a keyspace on NDB, named Specialized Self-Configuration Description (SSCD).

SSCM is a module within the SCE that acts only when, during the plug-and-play process, the MAC address of the element is matched to an SSCD. If not, the plug-and-play process normally occurs, considering that the connected element is a standard NE. But, if the physical address of the connected interface belongs to an element contained in this NDB table, a change in the SCE flow will be done, so that the specific configuration is made. This SSCD is implemented in JSON content-type format and contains the configuration characteristics for the network that will receive the element. The SSCD is added into the NDB through the NAD module, in the Administration Layer, which provides a friendly interface with the operator or system administrator.

The architecture of the SONAr system can be revised, including the SSCM module and the table with

the SSCDs, as shown in Figure 3. This figure complements the Figure 1, with the identification and location of the new modules within the layers that compose the SONAr.

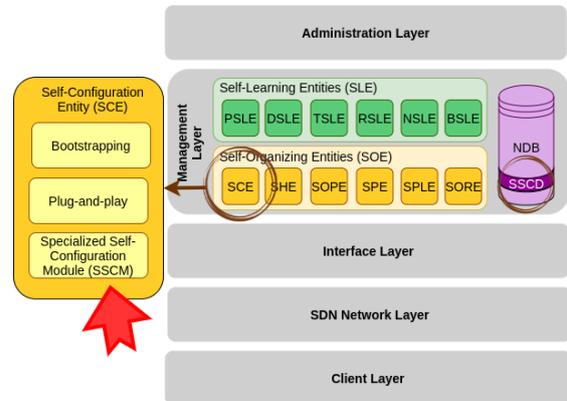


Figure 3: Simplified layered diagram of SONAr architecture showing SSCM module and SSCD database.

Note that SSCM is a module in SCE, together with bootstrapping and plug-and-play and SSCD is within NDB.

4.2 SSCM Operation

SSCM is a SONAr additional service for plug-and-play service: once active, when it detects that the inserted element is described by an SSCD in NDB, thus demanding a specific configuration, SSCM is started to carry out this process. The key that triggers it is the MAC address of the added edge element, which is also the key field in the NDB's specific keyspace, which may correspond to an SSCD. While plug-and-play continues to play its roles, configuring the natural routes between elements – enabling ARP and LLDP routing amongst other default signaling forwarding – SSCM takes care of more specialized configurations, such as packets with IEEE 802.1Q header (VLAN), prioritization in layer 2 (PCP), layer 3 (DiffServ), specific routes and other actions, depending on the specification described in the SSCD. These configurations are performed on all NEs in the path between the initial element (edge NE) and the final element identified on the SSCD (which is not necessarily the final destination element of the packets); additionally, specific flow rules are implemented to all elements in the path between these two NEs. Thus, SSCM assumes that the bootstrapping has already been performed (which is always true) and the basic plug-and-play configurations have already been implemented, so that SSCM is the last module to be executed within the SCE.

The insertion of SSCM in the context of self-

configuration in SONAr leads to some changes in the established flows of SCE. Essentially, two processes are changed to fit the SSCM module. These changes are shown in Figure 4.

The first change is in the Address Provider (AP), which is SONAr’s customized DHCP server: when receiving the DHCPDISCOVERY message, the AP first queries the NDB to check if there is an SSCD associated with the MAC address contained in the broadcast message. If so, in the DHCPOFFER message it offers the IP address and gateway address specified on the SSCD, instead of searching for a unreserved IP on NDB. This ensures that the edge NE can have its IP and gateway set by the engineer or network operator even before it is plugged into the network.

The second change occurs in the SCE: after generating the flow rules with the default routes, the SCE checks the MAC address of the inserted element’s interface. If it is in the SSCD base in NDB, SCE starts a specialized configuration process, generating specific flow rules. This process allows that, in addition to generic actions, e.g. simple default signaling route configurations, the SCE makes more elaborated configurations in the topology, as specified in the SSCD. After the generation of all flow rules is finished, they are all sent to the controller through NBI.

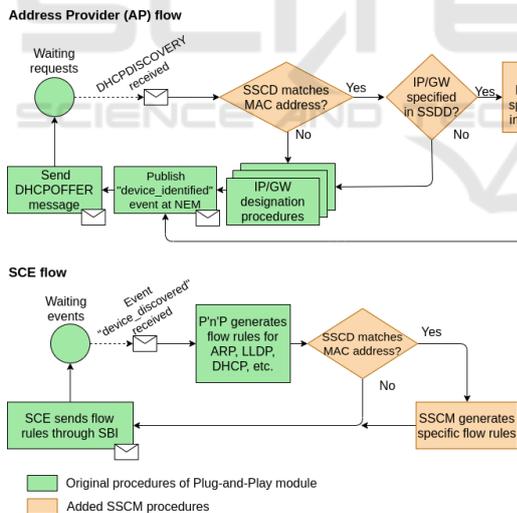


Figure 4: Message flow exchanges in SONAr elements by implementing SSCM.

4.3 SSCD Format and Functionalities

The entire process of self-configuration in SSCM begins with the specification of the specialized configuration to be made on the network so that it is prepared to connect a given edge element to its destination. This specification is performed through an object in JSON format containing the attributes describ-

ing the specificities of the network in question, the SSCD. This SSCD is a record in the NEBootConfiguration keyspace, on NDB. An example of an SSCD entry is shown below:

```

{
  "id": {
    "mac_address": "00:00:00:00:00:01",
    "description": "Edge NE",
    "ip": "192.168.0.100",
    "gateway": "192.168.0.200"
  },
  "config_end_node": {
    "mac_address": "00:00:00:00:00:02",
    "ip": "192.168.0.200",
    "description": "Provider edge router"
  },
  "dst_node": [
    {
      "description": "First Destination",
      "ip": "10.1.1.1",
      "priority": 5000,
      "vlan": 10,
      "pcp_cos": 1.
      ...
    },
    {
      "description": "Second Destination",
      ...
    }
  ]
}
    
```

The attributes of this SSCD object are interpreted as follows:

id: It is the identifier of the edge element to be automatically inserted in the network and the one that marks the starting point of the network configuration. It specifies the MAC address, which is the main identifier of the element (and which will be used to trigger the SSCM process) and other complementary information, such as the IP and gateway. If the IP and gateway identifiers do not exist, they will be defined and assigned by the AP, otherwise the AP will read these identifiers and send them in the DHCPOFFER message to the edge NE. The description and responsible identifiers only fulfill documentation roles.

config_end_node: It identifies the final element of the network’s self-configuration scope, that is, the self-configuration process is carried out in the path between the edge element and the element indicated in this field. This element is typically a gateway or an aggregator element. Its main identifier is also the MAC address of the outgoing port for the next element (which will not be automatically configured).

dst_node: It is an array containing the possible destinations of the packets sent by the edge element and the specification of what to do with them. The dst_node must be beyond config_end_node and may

be outside the SONAr domain. This is where the actions to be implemented in the network settings are characterized. For example, a packet sent from the edge element to a server (identified by the IP address) must travel on a specific VLAN with a given Class of Service (CoS). The priority field refers to the priority of the flow within the SDN context and not to the prioritization of the packet on the network, and it is recommended that it should be higher than those of the generic flow rules inserted by the plug-and-play (4,000 by default). The list of possible matching criteria and actions to be implemented is given by (ONOS Project, 2019). This array can contain as many target elements with specific actions as required, depending on the type of application.

4.4 Typical Application

The typical application for specialized self-configuration is in telecommunications networks, where it is compulsory to specify exactly the configuration domain and the parameters to be implemented. We can take as an example the integration of an eNodeB (4G cellular station) to a network of an MNO, as shown in Figure 5.

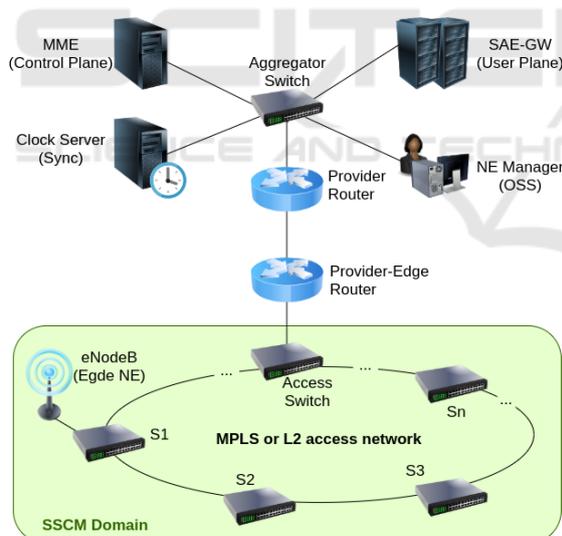


Figure 5: Example of a mobile network topology using SSCM for specialized configuration.

The eNodeB in Figure 5 is the start node identified by the id attribute on the SSCD, and is connected through an access network, usually an MPLS or a metro ethernet ring. This ring is integrated into the carrier's transport network by a provider edge router, and the network core elements are in a third network, accessed by a provider router. In the case of a cellular network, the target elements are the Mobility Management En-

tity (MME), the System Architecture Evolution Gateway (SAE-GW), the OSS server and the clock (or sync) server. These four core elements are the destination nodes, specified by the `dst_node` attribute on the SSCD.

For each of the four destinations, the first switch (S1) must, in order, insert an IEEE 802.1Q header containing the VLAN tag in the package, mark the package with a specific CoS, assign a priority to this flow rule (recommended to be higher than the priorities of the default flow rules which are 4,000 by default) and forward the packet on the correct outgoing interface. The other switches on the path to the provider edge router must forward the packets correctly and, finally, the last switch (access switch) must remove the IEEE 802.1Q header (and with it, the CoS mark) and forward the packet to the router. The same configuration procedure must be performed for packets coming from the core of the network to eNodeB: the access switch makes the markings while switch S1 removes them.

5 EXPERIMENTS

In this section, we present the experimentation performed in order to evaluate the overall performance of SSCM and discuss the results we have obtained.

5.1 Experimentation Environment

For emulation of the network environment, GNS3 software version 2.1.16 was chosen for several reasons. SONAr Server consists of all components of the SONAr system and the SDN Controller (ONOS 2.1.0); it was implemented in VM VirtualBox 6.0, OS Ubuntu Server 16.04 and developed using JDK 1.8.0-151. NDB was implemented with Cassandra 3.11.4 and NEM with RabbitMQ 3.8.0. The core and edge elements were implemented using Docker 18.06.2-ce containers, with OS Ubuntu 19.04. The switches and routers were based on Open vSwitch (OvS) 2.10.1 and implemented as containers on Docker Engine 18.06.2-ce, with OS Linux Alpine 3.9.

The topology used was as shown in Figure 6. In the access network (SSCM domain), different amounts of in-line switches were used to assess performance in each scenario.

5.2 Performance Evaluation

In this section, the experiments that evaluated the behavior of SSCM in networks with different amounts

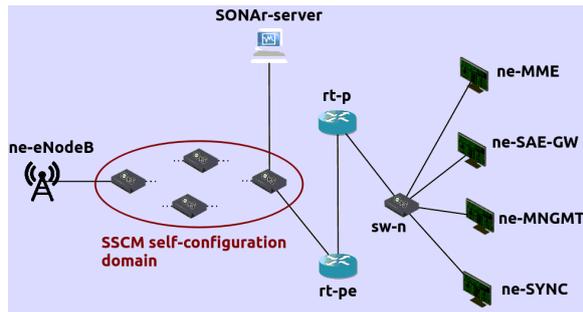


Figure 6: GNS3 topology adopted on experiments.

of elements in the access network will be discussed. Specifically, the scenarios for specialized self-configuration for 3, 6, 9, 12, 15 and 18 in-line switches were evaluated. All of these scenarios were simulated starting from the network bootstrapping to assess the impact of SSCM on the total self-configuration convergence time, starting from discovery until the final configuration of the network. Therefore, we have three stages:

- The discovery stage: when SONAr finds all network elements in its domain through the TCoE module, it establishes a topology in the form of a graph, records this topology in the NDB and notifies the other components of the completion of the process by publishing an event in the NEM.
- The generic configuration stage: it begins when the SCE receives the discovery completion event and the plug-and-play starts the configuration process, which involves calculating the shortest path between the SONAr server and the others network components and the generation of flow rules that will allow the communication of the elements through the control and signaling protocols (ARP, DHCP, LLDP, etc).
- The specialized configuration stage: an SSCD, corresponding to one of the network elements, is matched in the NDB. In this case, SSCM performs the calculation and generation of specialized flow rules for the network, after which all these flow rules (including the generic ones generated by the plug-and-play module) are sent to SDNC via NBI.

For this experiment, each scenario was simulated five times from the bootstrap and the times for each stage mentioned above were measured in milliseconds by a specific class, which recorded the intervals between the events received at the NEM. In this way, we have a very accurate measurement of the duration of the three stages of the network self-configuration. After measuring the times, the averages for each interval and for each scenario were calculated, and then the 95% confidence intervals were determined. All

graphs show the average along with the lower and upper limits of that range.

The results obtained for network discovery, generic configuration and specialized configuration are respectively shown in Figures 7, 8 and 9.

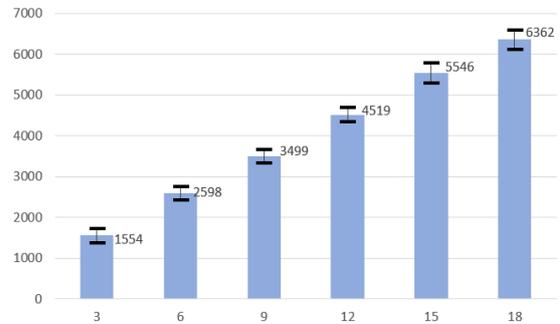


Figure 7: Time (in milliseconds) for network discovery for each in-line amount of switches.

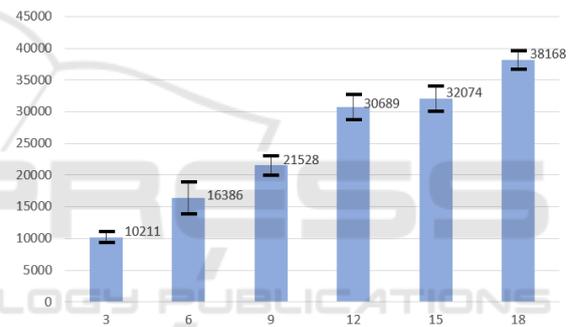


Figure 8: Time (in milliseconds) for network generic configuration for each in-line amount of switches.

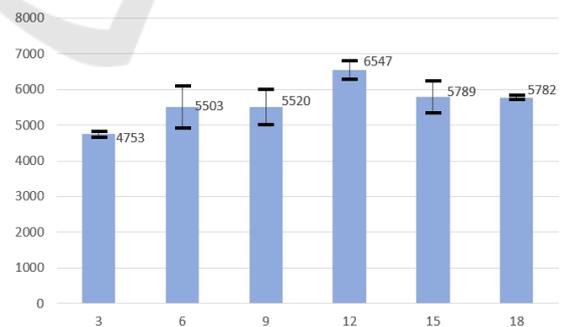


Figure 9: Time (in milliseconds) for network specialized configuration for each in-line amount of switches.

Looking only at the Figure 9, we notice that the SSCM convergence time does not vary substantially with respect to the number of switches that are in the path between the edge element and the provider edge router. This is explained by the fact that, when

the specialized configuration stage starts, all elements have already been identified, all routes have already been calculated and all generic configurations have been carried out, so that SONAr already has full knowledge of the network structure. There is, of course, an increase in the time involved in the calculation of the specific flow rules (the greater the number of elements, the longer the time of this calculation), however the duration of this process is very short when compared to the total self-configuration time, remaining at tens of milliseconds.

In discovery and generic network configuration, this relationship is much more obvious. In both cases, discovery and the configuration are done element by element in sequence, that is, it is necessary that one is done for another to start. This means that there is a direct and visually evident relationship between the convergence time of both stages and the number of elements in the path between the beginning and the end of the network.

In general, the time for the specialized configuration process – which is the one being evaluated in this study – was between 4 and 7 seconds for all samples obtained. It is noticeable that SONAr's response to an edge element addition in the network is fast enough for use in practice, exceeding by several orders of magnitude the time of manual configuration of a network even using semi-automated processes.

6 CONCLUSIONS

The Specialized Self-Configuration Module (SSCM), was proposed as a solution for the demand of network configuration automation. The main objective of the research was to investigate the alternatives for a network self-configuration system for elements with specific requirements, together with a model to describe these requirements. During the investigation, several approaches to this problem were studied, and the SONAr framework was well suited to the purpose. Within this architecture, a new module was designed and implemented (the SSCM) and a way to describe the specifics of the network configuration was proposed (the SSCD format).

In the experiments carried out, it was visible that the SSCM's performance far exceeds the estimated time for manual configuration, being a suitable approach to real life applications, specially in the telecommunications industry.

REFERENCES

- 3GPP (2018a). Telecommunication management; Self-configuration of network elements; Concepts and requirements. Technical Specification (TS) 32.501, 3rd Generation Partnership Project (3GPP).
- 3GPP (2018b). Telecommunication management; Self-Organizing Networks (SON); Concepts and requirements. Technical Specification (TS) 32.500, 3rd Generation Partnership Project (3GPP).
- Bianchi, G. and *et al.* (2016). Superfluidity: a flexible functional architecture for 5G networks: G. Bianchi *et al.* *Transactions on Emerging Telecommunications Technologies*, 27(9):1178–1186.
- Chiosi, M., Clarke, D., Willis, P., Reid, A., Ferger, J., Bugenhagen, M., Khan, W., Fargano, M., Cui, C., Deng, H., Benitez, J., and *et al.* (2012). Network Functions Virtualisation, An Introduction, Benefits, Enablers, Challenges & Call for Action.
- Ganek, A. G. and Corbi, T. A. (2003). The dawning of the autonomic computing era. *IBM Systems Journal*, 42(1):5–18.
- Gonçalves, M., Souza Neto, N., Oliveira, D., Silva, F., and Froisi, P. (2020). Bootstrapping and plug-and-play operations on software defined networks: A case study on self-configuration using the sonar architecture. In *Proceedings of the 10th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER.*, pages 103–114. INSTICC, SciTePress.
- Jiang, W., Strufe, M., and Schotten, H. D. (2017). A SON decision-making framework for intelligent management in 5G mobile networks. In *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pages 1158–1162, Chengdu. IEEE.
- Katiyar, R., Pawar, P., Gupta, A., and Kataoka, K. (2015). Auto-Configuration of SDN Switches in SDN/Non-SDN Hybrid Network. In *Proceedings of the Asian Internet Engineering Conference on - AINTEC '15*, pages 48–53, Bangkok, Thailand. ACM Press.
- Neves, P., Calé, R., Costa, M. R., Parada, C., Parreira, B., Alcaraz-Calero, J., Wang, Q., Nightingale, J., Chirivella-Perez, E., Jiang, W., Schotten, H. D., Koutsopoulos, K., Gavras, A., and Barros, M. J. (2016). The SELFNET Approach for Autonomic Management in an NFV/SDN Networking Paradigm. *International Journal of Distributed Sensor Networks*, 12(2):2897479.
- Oliveira, D., Souza Neto, N., Gonçalves, M., Silva, F., and Froisi, P. (2020). Network self-configuration for edge elements using self-organizing networks architecture (sonar). In *Proceedings of the 10th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER.*, pages 408–414.
- ONOS Project (2019). Flow Rules - ONOS. Available at wiki.onosproject.org/display/ONOS/Flow+Rules.
- Open Networking Foundation (2012). Software-defined networking: The new norm for networks. White paper.
- TELUS and Huawei (2016). Next generation son for 5g.