Optimized Provisioning Techniques for Geo-distributed SDP-enabled Next Generation Networks Security

1st Yahuza Bello
Department of Computer Engineering
University of Guelph
Guelph, Canada
ybello@uoguelph.ca

2nd Ahmed Refaey
Department of Computer Engineering
University of Guelph
Guelph, Canada
ahmed@uoguelph.ca

3rd Petros Spachos
Department of Computer Engineering
University of Guelph
Guelph, Canada
petros@uoguelph.ca

ICC 2023 - IEEE International Conference on Communications | 978-1-5386-7462-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICC45041.2023.10279552

***Abstract*—What advancements might Next Generation Networks (NGN) unleash that existing ones cannot? NGNs are envisioned to empower the connection between billions of people and zillions of heterogeneous Internet of Things (IoT) devices while consolidating intelligence and autonomy. However, several security-related challenges will be introduced and apparently, the security solutions and architectures used in previous network generations will not be sufficient. The Cloud Security Alliance's (CSA) Software Defined Perimeter (SDP) is a potential candidate to provide the much-needed security framework for next-generation networks. However, the lack of a scalable SDP controller will be a considerable drawback for the wide adoption of the SDP framework. Therefore, this paper focuses on modeling a multi-SDP controller placement problem as a VNF-FGE in a Geo-distributed NFV-based environment as a potential solution to secure next-generation networks. Due to its NP-hard nature, this type of problem can be addressed by extending the NCO approach via Reinforcement Learning (RL) to optimize the reward policy in accordance with the constraints of the problem. The agent developed can learn the placement decisions of the SDP controllers by inference (i.e., policy strategy) through the RL process. The experiment's analysis reveals the RL approach's superiority over the well-known Gecode optimization solver.**

***Index Terms*—Reinforcement learning, Optimization, SDP, Scalability**

I. Introduction

Next-generation networks are being deployed and advanced communication technologies are coming to market. This improves the number of data transmission speeds, the number of cells, and the latency significantly [1]. In order for next-generation networks to achieve the computing, security, and infrastructure requirements, technologies such as Artificial Intelligence (AI), Software Defined Networking (SDN), and Network Function Virtualization (NFV) will likely be integrated into all network applications at various levels. However, several security-related challenges will be introduced and apparently, the security solutions and architectures used in previous network generations will not be sufficient for the next-generation networks. Furthermore, the development of new services and technologies in the next-generation networks will require new security solutions, and architectures [2].

Due to the sophisticated nature of the next-generation networks (i.e., enabling the rise of new services and devices with new requirements such as low latency and high throughput), multiple technologies are required to design a new reference architecture. For example, the Next Generation Mobile Network's (NGMN) 5G design principles emphasize the importance of highly elastic and robust systems [3]. With the common composable core network, for example, user planes and control planes will be separated and dynamic placement of network functions will be enabled through SDN and NFV [4]. Additionally, the next-generation network's architecture must allow for the deployment of virtualized security modules within the network perimeter. Thanks to NFV technology, this can be achieved by deploying security modules as Virtual Network Functions (VNF) within the network. However, these enabling technologies (i.e., NFV, SDN, AI, etc) have security vulnerabilities that must be addressed to be fully utilized in the next-generation networks.

One of the most challenging issues in many of the proposed security frameworks in the literature is the scalability feature of the modules in the security framework. Whenever network infrastructure is scaled out (e.g., mobile core network entities) to accommodate for increment in users, the security framework used in the network becomes overwhelmed and often results in exposing the network to many cyber attacks. Therefore, the need for scalable software-based security frameworks rises rapidly to tackle this issue. Software Defined Perimeter (SDP) is an excellent candidate to provide the much-needed scalable security framework [5]. SDP secures network services using a zero-trust model, in which all the components involved require to be verified and authenticated by the SDP controller before having access to any of the protected network services.

Several research works in the literature have recently explored SDP as a potential security solution in various applications [6]–[9]. However, none of them have explored the scalability issues

of SDP components (specifically the SDP controller). In this work, we aim to solve this issue by modeling the scalable SDP controller problem as a VNF Forward Graph Embedding (VNF-FGE) problem.

Placement optimization problems (specifically VNF-FGE) being constrained combinatorial optimization problems are proven to be NP-hard in the literature [10]. As a result, an exact or optimal solution can only be derived for small-scale scenarios because the computational cost of the optimal solution is not affordable in large-scale scenarios. To balance the optimality and computational cost trade-off, a large number of existing approaches in the literature focus on heuristic algorithms or metaheuristic algorithms. Although these approaches yield good results under reasonable assumptions, they tend to be ineffective in a highly constrained environment and require both high expertise and domain knowledge to design. It has been found that Neural Combinatorial Optimization (NCO) [11]is a useful method for solving combinatorial optimization problems by utilizing Reinforcement Learning (RL) Traveling Salesman Problem (TSP) [12] and Vehicle Routing Problem (VRP) [13] are examples of classical combinatorial problems where this approach has achieved near-optimal solutions. In particular, the NCO model estimates the relationship between the instances of the problem and their solutions by establishing a Neural Network (NN) model. Specifically, an RL approach is used to estimate the parameters of the model iteratively.

Therefore, in this paper, we focus on modeling a multi-SDP controller placement problem as a VNF-FGE in a Geo-distributed NFV-based environment as a potential solution to securing next-generation networks. This type of problem can be addressed by extending the NCO paradigm and optimizing the reward policy in accordance with the constraints of the problem. To achieve this, we extend the VNF placement optimization in [14] taking into account the added security constraints applicable to the SDP controller. In particular, a sequence-to-sequence model is implemented that, for a particular authentication and authorization request, applies a placement policy that minimizes overall power consumption while considering the state of the NFV infrastructure. The detailed problem formulation will be presented in section IV.

The rest of the paper is outlined as follows: Section II presents the related works in SDP and NCO. section III introduces the specifications of the multiple SDP controllers within the NFVI architecture. Section IV introduces the proposed optimized multi-SDP controller placement problem. An optimization model is formulated and explained in detail. Then the optimization policy strategy using the RL approach is introduced and explained. The performance evaluation of the proposed optimized SDP controller model is presented in Section V. Section VI presents our concluding remarks.

II. Related Works

Several research works in the literature have recently explored SDP as a software-based security solution. SDP was recently proposed to secure Infrastructure as a Service (IaaS) [15]. The authors implement SDP in an Amazon Web Service (AWS) environment and verify its resilience against DoS and port scanning attacks. In [6], an SDP-MEC architecture was proposed to secure devices at the edge of the network. In order to prevent various cyber attacks, SDP entities (e.g., the SDP gateway and SDP controller) were placed at the edge of the network. As demonstrated in [16], SDP controllers can be integrated with SDN controllers in a combined SDN-SDP architecture to address the security risks posed by the abstraction of the data plane and control plane in SDN. SDP was also proposed in an NFV environment to secure VNFs deployed in Network Function Virtualization Infrastructure (NFVI) [7]. The SDP entities were virtualized and deployed as VNFs in the NFVI to darken it against attacks by intruders. This idea is extended in [8] to develop a virtual Evolved Packet Core - virtual Software Defined Perimeter (vEPC-vSDP) framework. By utilizing the authentication and authorization mechanism of the SDP, the mobile core network entities were able to communicate securely via the proposed framework. The SPA-based authentication approach of the SDP framwork has been adopted in Message Queuing Telemetry Transport (MQTT) [17]. Recently, the scalability, reliability, and usability-related issues of the SDP framework was investigated [9]. The authors proposed multiple architectures for a scalable SDP controller such as hierarchical, bridge, and hybrid models. However, only qualitative evaluations of the models were presented.

It is clear that none of the existing research studies that adopt the SDP framework consider the scalability issue that will eventually arise in case of either a failed component (mainly the SDP controller) or an increase in users in the network that adopts SDP as a security framework. This opens a serious challenge in the wide adoption of SDP as a security framework in next-generation networks. Therefore, this paper proposes multi-SDP controller placement as a VNF-FGE optimization problem as explained earlier.

Several research studies have explored NCOs in solving combinatorial optimization problems. In [18], the authors propose Multi-Agent Reinforcement Learning (MARL) as a solution for maximizing both overall and fairness throughput using Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs). Leveraging RL and contrastive self-supervised learning, the authors in [19] propose a novel method for solving two combinatorial optimization problems: TSP and Capacity Vehicle Routing Problem (CVRP). The authors of [20] showed that supervised learning can be utilized to solve combinatorial optimization problems. Specifically, they demonstrated that a NN can return near-optimal solutions to problems given suitable inputs. The authors in [21] proposed a methodology to learn device placement decisions in the domain of TensorFlow computational graphs. Their methodology depends on the use of a sequence-to-sequence model for prediction. The model infers the devices that will run which operation subsets in a

TABLE I: Table of notation

Notation	Definition
H	Set of hosts with index $h \in H$
X	Set of SDP controllers with index $i \in X$
L	Set of links with index $l \in L$
R	Set of resources with index $r \in R$
$r_{r,i}$	Amount of r resources requested by SDP controller $i \in X$
$a_{r,i}$	Amount of r resources available in controller $i \in X$
P_h^{CPU}	Power consumption in terms of CPU in host server $h \in H$
P_h^{min}	Idle state power consumption in terms of CPU in host server $h \in H$
P_{net}	Power consumption in terms of bandwidth utilization on link $l \in L$
B_l	Total bandwidth of link $l \in L$
b_i	Bandwidth requested by SDP controller $i \in X$
P_i	Processing capacity of SDP controller $i \in X$ in terms of authentication process
δ_i	Search time for authentication keys of each SDP gateway and SDP clients for SDP controller $i \in X$
l_i	latency on the link $l \in L$ caused due to SDP controller $i \in X$
L^{max}	Maximum latency allowed on the link $l \in L$

TensorFlow graph.

The architecture proposed in this work is depicted in Figure 1. It consists of several VNFs hosting multiple SDP controllers (four SDP controllers in this example) and other network functions in an NFV-based environment. Network functions are placed within multiple Network Function Virtualization Infrastructure - Point of Presences (NFVI-PoPs) as shown in Figure 1. Note that the Virtual Infrastructure Manager (VIM) handles the underlying physical networking (i.e., in this case, the VNFs are attached to physical switches). The main goal here is to deploy multiple SDP controllers in different servers within either a single NFVI-PoP or multiple NFVI-PoP. In this way, there will be redundant SDP controllers in standby mode in case of a failed SDP controller.

To further illustrate this scenario, let us assume we have multiple SDP controllers to be deployed within the NFV environment. Each of the available servers within NFVI-PoPs has different specifications in terms of CPU, memory, and storage capacities. To simplify the networking, we assume the host servers are connected in a star topology similar to the work in [14]. Now, the problem is to optimally place the SDP controllers in the hosts servers, minimizing some overall cost functions and satisfying some desired constraints. The detailed optimization formulation will be presented in section IV.

III. Optimized Multi-SDP controller Placement

A. Problem Formulation

We assume a distributed network consisting of multiple nodes hosting SDP controllers, SDP gateways, SDP clients, and forwarding nodes. As stated earlier, all SDP components require to be authenticated by the SDP controller. This will eventually lead to the exhaustion of the controller's processing capacity (in terms of time allocated for searching for authentication keys) in a large network with hundreds of thousands of SDP gateways and clients. Additionally, having a single SDP controller will lead to a single point of failure, which will expose the network to multiple cyber attacks. Therefore, the ultimate goal is to deploy the SDP controllers to multiple controller domains to serve the authentication and authorization purposes of the SDP gateways and clients.

To achieve the above mentioned objective, we model the problem as an Integer Linear Programming (ILP) problem. The objective of this optimization formulation is to minimize the overall power consumption of the physical NFVI, which takes into account the aggregated sum of the power consumption of the servers hosting the SDP controllers and the cost of the activated links used within the SDP framework as illustrated in (4).

Note that we assume a network, in which its topology, links between its nodes, and the deployment of the client and the gateway are known. The optimization model's job is to deploy the SDP controllers within the network taking into account multiple constraints shown in (5)-(9).

$$\phi_{i,h} = \begin{cases} 1, & \text{if SDP controller } i \in X \text{ is hosted in } h \in H \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

$$\psi_h = \begin{cases} 1, & \text{if host server } h \in H \text{ is activated} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

$$\chi_h = \begin{cases} 1, & \text{if link } l \in L \text{ is activated} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

$$min_{\phi_{i,h}} \Big(\sum_{h \in H} \Big[P_h^{CPU} \sum_{i \in X} r_{r,i} \times \phi_{i,h} + P_h^{min} \times \psi_h \Big] + \sum_{l \in L} P_{net} \sum_{i \in X} b_i \times \phi_{i,h} \Big) \tag{4}$$

$$s.t. \quad \sum_{i \in X} \phi_{i,h} \leq 1 \qquad \forall h \in H \tag{5}$$

$$\sum_{i \in X} \delta_i \times \phi_{i,h} \leq P_i \qquad \forall h \in H \tag{6}$$

$$\sum_{i \in X} r_{r,i} \times \phi_{i,h} \leq \psi_h \times a_{r,i} \qquad \forall h \in H, r \in R \tag{7}$$

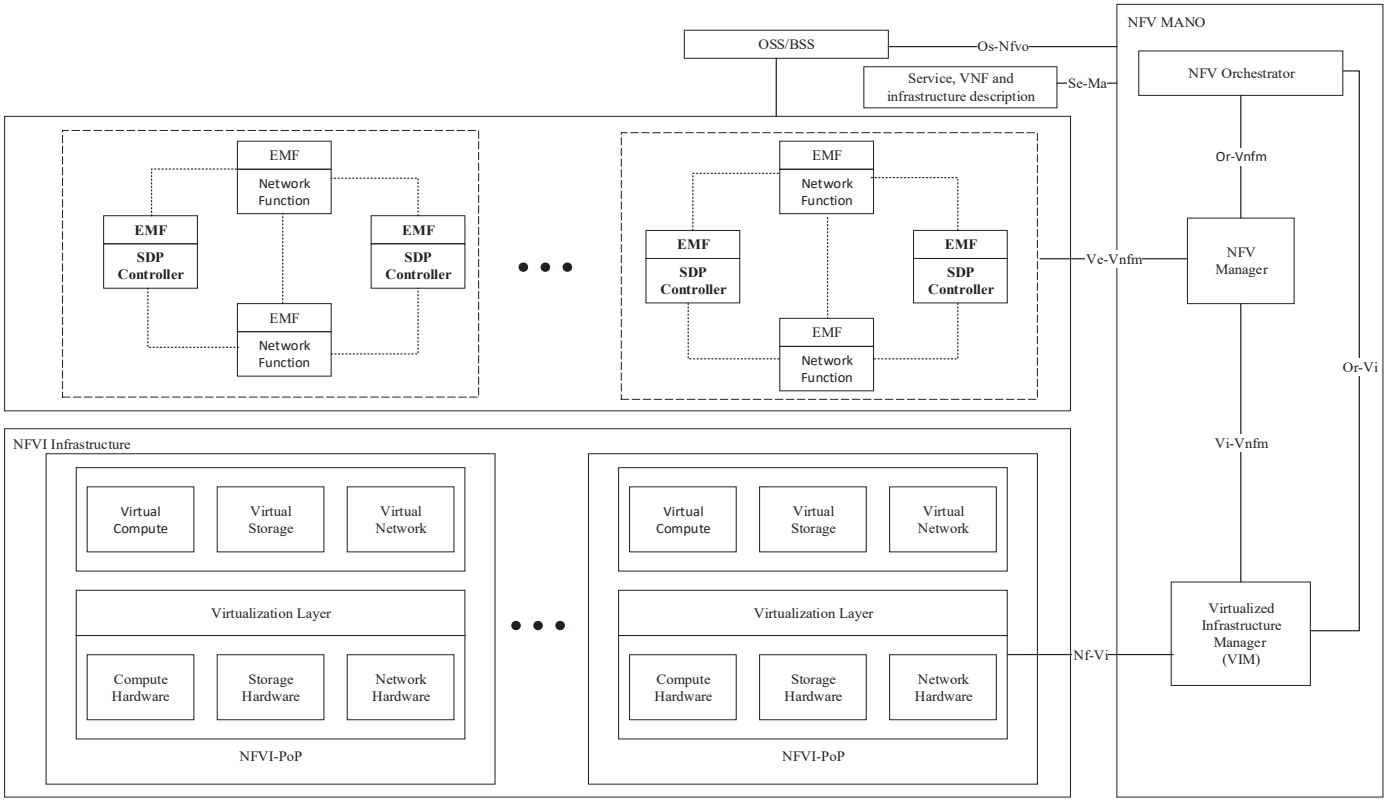

Fig. 1: Multiple SDP controllers placement within the NFVI

$$\sum_{i \in X} b_i \times \phi_{i,h} \leq \chi_h \times B_l \qquad \forall h \in H, l \in L \tag{8}$$

$$\sum_{l \in L} \sum_{i \in X} l_i \times \phi_{i,h} \leq L^{max} \qquad \forall h \in H \tag{9}$$

$$\phi_{i,h} \in \{0,1\} \qquad \forall i \in X, h \in H \tag{10}$$

$$\psi_{i,h} \in \{0,1\} \qquad \forall i \in X, h \in H \tag{11}$$

$$\chi_h \in \{0,1\} \qquad \forall i \in X, h \in H \tag{12}$$

We consider the network topology as an undirected graph $G = (H, L)$, where H represents the set of nodes (i.e., host servers in this case) and L represents the set of physical links in the network. The set of host servers is denoted by H with index $h \in H$. The set of SDP controllers is denoted by X with index $i \in X$. The set of links is denoted by L with index $l \in L$. The set of available resources in terms of CPU capacity is denoted by R with index $r \in R$. We denote the amount of r available resources in controller $i \in X$ as $a_{r,i}$. The amount of power consumption in terms of CPU in the host servers $h \in H$ is represented as P_h^{CPU} while the amount of idle power consumption in terms of CPU in the host servers $h \in H$ is represented as P_h^{min}. Likewise, the power consumption in terms of bandwidth utilization on link $l \in L$ is denoted by P_{net}. The total bandwidth of link $l \in L$ is given as B_l. The bandwidth requested by SDP controller $i \in X$ is represented as b_i. The processing capacity of the SDP controller $i \in X$ in terms of maximum authentication processes handled is represented as P_i. The time used to search authentication keys in the MySQL table of by SDP controller $i \in X$ to authenticate SDP gateways and SDP clients is represented as δ_i. The latency on the link $l \in L$ caused due to SDP controller $i \in X$ is represented by l_i and the maximum allowable latency on the link $l \in L$ is represented as L^{max}. In Table I, a summary of the notations and the descriptions of the parameters used in the formulation of the problem is presented.

The objective function as explained earlier is to minimize the overall power consumption of the physical NFVI, which takes into account the aggregated sum of the power consumption of the servers hosting the SDP controllers and the cost of the activated links used within the SDP framework, which is represented in (4). $\phi_{i,h}$ is the binary decision variable that controls the placement decision of SDP controller $i \in X$ as shown in (1). The variable $\psi_{i,h}$ is used to control the activation and deactivation of host servers in the infrastructure as shown in (2). When a host server is activated, the value is set to 1

otherwise it is set to 0. Likewise, the variable χ_h is used for activated and deactivated links as shown (3). It is 1 when a link is active, which means that the link is carrying traffic else it is 0 for inactive links.

Constraint (5) ensures that any SDP controller can only be hosted on one host server. Constraint (6) ensures that each SDP controller does not exceed the maximum processing capacity in terms of authentication procedures. In order to achieve equilibrium, constraint (7) specifies that the entire amount of resources used in an active server cannot go above the number of resources available in it. With this constraint, only the host servers (i.e., the servers where the SDP controllers are deployed) are considered active. Constraint (8) deals with bandwidth limitation, in which the reserved bandwidth for the SDP controller does not exceed the available bandwidth of the link. Note that we use the link activation variable in the same fashion as in the host activation. In constraint (9), the link latency requirement is established. Constraint (10) declares a binary decision variable. Constraint (11) and (12) are the two binary supporting variables.

B. Optimization Policy Using RL Approach

Due to the limitations of heuristics and metaheuristics algorithms, in this work, we aim to utilize the NCO approach to solve the SDP controller placement problem. For inferring a placement policy, we use a NN model based on encoder-decoder structures similar to the work in [14],. The encoder-decoder structure is based on a sequence-to-sequence model, which has yielded excellent results in sequence-prediction.

In short, the RL process works as follows. An agent receives as inputs a set of SDP controllers, X to be placed in the NFV-based environment and outputs a placement vector indicating the host's servers. The weights of the NN imply a placement strategy $\pi_\theta(p^c|x)$ for all possible combinations of the SDP controller placements. This means that, for a specific SDP controller placement request, a state vector, which consists of the environment state and the requested placement is created to input to the agent. As a result, the agent computes a set of placement decisions that specifies the new deployment locations of the SDP controllers. In order to calculate the reward, the environment evaluates the placement decision based on (4). Note that the environment described by (1)-(12) to the agent is considered a black box, which the agent uses to infer a policy that provides a solution to the SDP controller placement problem.

One of the limitations of the NCO approach is the fact that it is applied to provide solutions to unconstrained optimization problems. To overcome this issue, we adopt the constrained relaxation technique in the cost function introduced in [14]. To this end, additional feedback signals are used to identify the degree of constraint dissatisfaction alongside the reward signal that indicates the overall infrastructure's power consumption.

To learn the parameters of the Stochastic Policy $\pi_\theta(p^c|x)$, we utilize the policy gradient technique, which uses the input placement request to assign certain probabilities to specific placements (i.e., a placement with high probability exhibits lower cost and a placement with low probability exhibits higher cost). The NN uses (13), which is a chain rule to factorize the output probabilities [**?**]. Note that $p(< i)$ represents the previously allocated VNFs. This means that the probability of assigning a VNF in the sequence depends on the previously assigned VNFs and the state vector. Similar to [14], we assume the state vector is predefined.

$$\pi_\theta(p^c|x) = \prod_{i \in X} \pi_\theta(p_i|p(< i), x) \tag{13}$$

In an NCO approach that utilizes the policy gradient technique, the objective function represents the expected reward that can be gained for each vector of weight θ, which determines the optimum policy desired as shown in (17) [14]. Now the main target is to discover a policy that minimizes the overall expected energy consumption subject to a defined constraint.

$$\begin{aligned} &min_{\pi \sim \Pi} J_E^\pi(\theta) \\ &\text{s.t.} \quad J_{Dj}^\pi \leq 0 \end{aligned} \tag{14}$$

Where $J_E^\pi(\theta)$ is the expected energy cost as shown in (15) [14]. Note that $J_E^\pi(\theta|x)$ represents the expected energy consumption, E associated with a placement request as shown in (16) [14]. J_{Dj}^π represents the constraint dissatisfaction signal returned by the environment. This indicates the accumulated constraint dissatisfaction, which are grouped according to occupancy, bandwidth, and latency similar to [14]. The constraint dissatisfaction signal with respect to a policy is defined in (17) [14].

$$J_E^\pi(\theta) = \mathbb{E}_{x \sim X}[J_E^\pi(\theta|x)] \tag{15}$$

$$J_E^\pi(\theta|x) = \mathbb{E}_{p \sim \pi_\theta(.|x)}[E(p)] \tag{16}$$

$$J_D^\pi(\theta) = \mathbb{E}_{x \sim X}[J_D^\pi(\theta|x)] \tag{17}$$

As stated earlier, the NCO approach is applicable to only unconstrained optimization problems. Therefore, the optimization presented in (14) is converted to an unconstrained optimization problem using the Lagrange relaxation technique as shown in (18).

$$\begin{aligned} g(\lambda) &= min_\theta J_L^\pi(\lambda, \theta) \\ &= min_\theta [J_E^\pi(\theta) + \sum_i \lambda_j \times J_{Dj}^\pi] \\ &= min_\theta [J_E^\pi(\theta) + J_E^\pi(\rho)] \end{aligned} \tag{18}$$

Where $J_L^\pi(\lambda, \theta)$ is the Lagrange cost function, $g(\lambda)$ is the Lagrange dual function and λ_j are the Lagrange multipliers. Similar to [14], an additional function, for the expected penalization $J_E^\pi(\rho)$ is used.

Stochastic gradient descent and Monte-Carlo Policy Gradients

are used to calculate weights θ that optimize the objective function in (18) using (19) - (22) as follows:

$$\theta_{n+1} = \theta_n + \alpha \times \nabla_\theta J_L^\pi(\theta) \tag{19}$$

Where $\nabla_\theta J_L^\pi(\theta)$ is the gradient of the Lagrangian [22] computed as shown in (20) and (21). Note that the term $L(p|x)$ represents the penalty induced in the energy cost in each iteration.

$$\begin{aligned} \nabla_\theta J_L^\pi(\theta) &= \mathbb{E}_{p\sim\pi_\theta(.|x)}[L(p|x) \times \nabla_\theta \log \pi_\theta(p|x)] \\ L(p|x) &= E(p|x) + \rho(p|x) \\ L(p|x) &= E(p|x) + \sum_i \lambda_i \times D_i(p|x) \end{aligned} \tag{20}$$

Finally, the gradient is estimated using the Monte-Carlo sampling technique [**?**], where B samples are picked randomly from the sample space. In order to reduce the variance of the gradient, a baseline estimator $b(x)$ is added; this will result in faster convergence [**?**].

$$\nabla_\theta J_L^\pi(\theta) \approx 1/B \sum_{j\in B} (L(p_j|x_j) - b_{\theta_v}(x_j)) \times \nabla_\theta \log \pi_\theta(p_j|x_j) \tag{21}$$

$$\zeta(\theta_v) = 1/B \sum_{j\in B} ||b_{\theta_v}(x_j) - L(p_j|x_j)||^2 \tag{22}$$

IV. Experiment And Analysis

In order to evaluate the RL approach used in solving the SDP controller placement problem, we utilized the VNF placement implementation provided in [14]. We added the additional constraints required to adapt the implementation to suit our placement problem. The parameters related to overall power consumption within the environment are taken as follows: $P_h^{CPU} = 100$ Watt, $P_h^{min} = 200$ Watt, and $P_{net} = 0.1$ Watt/mbps. We evaluate the performance of the NN network against the Gecode solver on a host machine running Linux Ubuntu 18.04.3 LTS Bionic Beaver. The specification of the testbed and the NN is summarized in Table II and Table III respectively. Note that the number of Long Short-Term Memory (LSTM) layers and the number of LSTM hidden layers correspond to each other. That is, 1 layer corresponds to 32 hidden layers, and so on.

We begin by analyzing the learning process of the sequence-to-sequence model as shown in Figure 2. In each iteration, an approximation for baselines, penalties, and Lagrangian functions is demonstrated. It is clear that a high penalty is received by the agent for violating many constraints at the beginning of the learning process. This means that, at the beginning of the learning process, the agent ignores the underlying power consumption objective in favor of constraint satisfaction. However, as the learning proceeds, the agent minimizes its Lagrangian objective function (i.e., (18) via stochastic gradient descent. This process is iterated to continuously improve the agent's policies.

TABLE II: Environment Specifications

Properties	Value					
Number of CPUs (core)	10	10	9	9	8	8
Link bandwidth (Mbps)	1000	500	1000	500	400	300
Link latency (ms)	50	10	50	50	50	50

TABLE III: Neural Network characteristics

Hyperparameters	Value
Agent's learning rate	0.0001
Baseline's learning rate	0.01
Number of LSTM layers	1, 3, 4
Number of LSTM hidden layers	32, 64, 128
Embedding size	10
Number of model inference	6

One interesting trend in the result shown in Figure 2 is the penalty approximation. As the learning process progresses over many iterations, this signal goes toward zero. This is because the agent learns a good policy for the SDP controller placement by satisfying the constraints. Thus receiving very small penalization for constraint dissatisfaction.

In Figure 3, the placement error ratio percentages are presented for the RL approach and the Gecode solver. Overall, both techniques show similar trends with the Gecode solver performing slightly worse than the RL approach when the number of SDP controllers hosted per placement exceeds 70. This means that as the number of SDP controllers to be placed increases (in other words the instances of the optimization problem increase), the RL approach has a better chance of finding feasible solutions within the solution space. This further validates the advantage of using NCO to solve NP-had problems for large-scale scenarios.

V. Conclusion

This paper presents a VNF-FGE in the form of optimization for the SDP controller placement problem. SDP has shown great resilience against several cyber attacks as many authors have demonstrated in the literature. However, the lack of multiple SDP controllers is bound to cause several hiccups for the wide adoption of the SDP framework. This work formulates the SDP controller placement problem and utilizes the NCO approach to solve it. For that purpose, we adopt the extended version of the NCO to consider the constraints in the SDP controller placement problem. The main goal is to host multiple SDP controllers while minimizing the overall power consumption of the NFVI. The neural network was able to learn the placement decisions of the SDP controllers by selecting the correct policy through the RL process. The experimental results demonstrated the superiority of the RL approach over the well-known Gecode optimization solver.

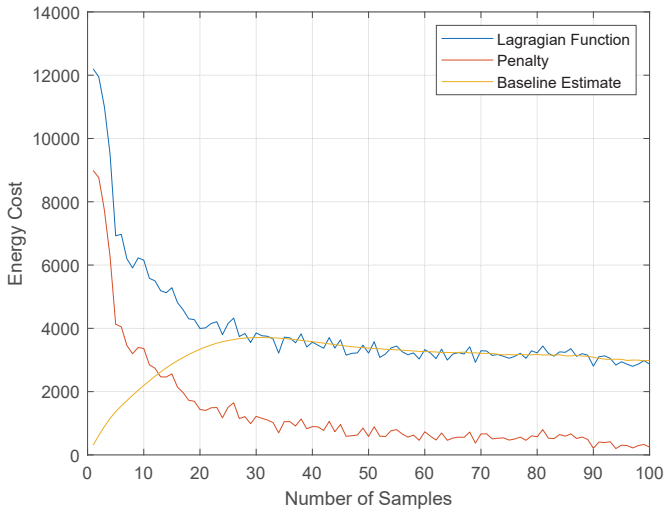

Fig. 2: Learning history

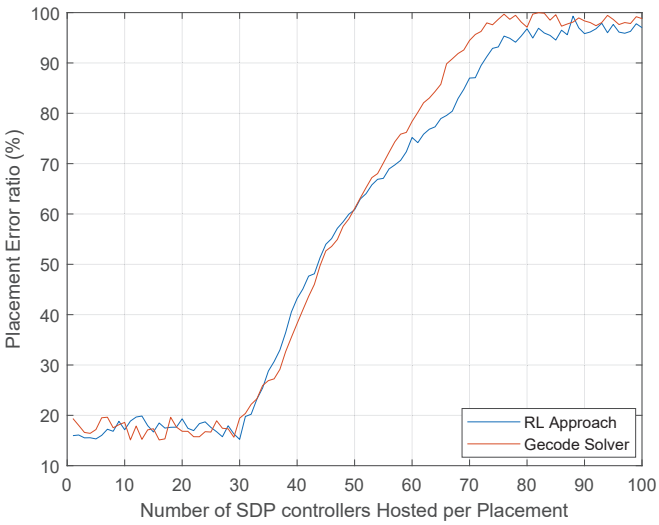

Fig. 3: Placement error ratio for multiple SDP controller placement

REFERENCES

[1] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, "The road towards 6g: A comprehensive survey," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, 2021.

[2] G. Arfaoui, P. Bisson, R. Blom, R. Borgaonkar, H. Englund, E. Félix, F. Klaedtke, P. K. Nakarmi, M. Näslund, P. O'Hanlon *et al.*, "A security architecture for 5g networks," *IEEE Access*, vol. 6, pp. 22 466–22 479, 2018.

[3] "Ngmn 5g white paper." [Online]. Available: https://www.ngmn.org/wp-content/uploads/NGMN_5G_White_Paper_V1_0.pdf

[4] X. Costa-Perez, A. Garcia-Saavedra, X. Li, T. Deiss, A. De La Oliva, A. Di Giglio, P. Iovanna, and A. Moored, "5g-crosshaul: An sdn/nfv integrated fronthaul/backhaul transport network architecture," *IEEE wireless communications*, vol. 24, no. 1, pp. 38–45, 2017.

[5] "Sdp: The most advanced zero trust architecture," Software Defined Perimeter Working Group, May 2020. [Online]. Available: https://cloudsecurityalliance.org/artifacts/sdp-the-most-advanced-zero-trust-architecture/

[6] J. Singh, Y. Bello, A. Refaey, A. Erbad, and A. Mohamed, "Hierarchical security paradigm for iot multi-access edge computing," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[7] J. Singh, A. Refaey, and A. Shami, "Multilevel security framework for nfv based on software defined perimeter (sdp)," *IEEE Network*, pp. 1–6, March 2020.

[8] Y. Bello, A. Refaey, M. Ulema, and J. Kolipali, "On sustained zero trust conceptualization security for mobile core networks in 5g and beyond," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[9] Y. Palmo, S. Tanimoto, H. Sato, and A. Kanai, "A consideration of scalability for software defined perimeter based on the zero-trust model," in *2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2021, pp. 717–724.

[10] S. Agarwal, F. Malandrino, C. F. Chiasserini, and S. De, "Vnf placement and resource allocation for the support of vertical services in 5g networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 1, pp. 433–446, 2019.

[11] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016. [Online]. Available: https://arxiv.org/abs/1611.09940

[12] Z. Zhang, H. Liu, M. Zhou, and J. Wang, "Solving dynamic traveling salesman problems with deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021.

[13] Y. Xu, M. Fang, L. Chen, G. Xu, Y. Du, and C. Zhang, "Reinforcement learning with multiple relational attention for solving vehicle routing problems," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 11 107–11 120, 2022.

[14] R. Solozabal, J. Ceberio, A. Sanchoyerto, L. Zabala, B. Blanco, and F. Liberal, "Virtual network function placement optimization with deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 292–303, 2020.

[15] J. Singh, A. Refaey, and J. Koilpillai, "Adoption of the software-defined perimeter (sdp) architecture for infrastructure as a service," *Canadian Journal of Electrical and Computer Engineering*, vol. 43, no. 4, pp. 357–363, 2020.

[16] A. Sallam, A. Refaey, and A. Shami, "On the security of sdn: A completed secure and scalable framework using the software-defined perimeter," *IEEE Access*, pp. 1–1, 2019.

[17] A. Refaey, A. Sallam, and A. Shami, "On iot applications: a proposed sdp framework for mqtt," *Electronics Letters*, 09 2019.

[18] S. Yin and F. R. Yu, "Resource allocation and trajectory design in uav-aided cellular networks based on multiagent reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2933–2943, 2022.

[19] Z. Yuan, G. Li, Z. Wang, J. Sun, and R. Cheng, "Rl-csl: A combinatorial optimization method using reinforcement learning and contrastive self-supervised learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–15, 2022.

[20] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper/2015/file/29921001f2f04bd3baee84a12e98098f-Paper.pdf

[21] A. Mirhoseini, H. Pham, Q. V. Le, B. Steiner, R. Larsen, Y. Zhou, N. Kumar, M. Norouzi, S. Bengio, and J. Dean, "Device placement optimization with reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2430–2439.

[22] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.