Efficient Provisioning of On-Demand Security Services using Chained Virtual Security Network Functions

Godwin Thomas Samuel Chapanduka

Department of Electrical Engineering, Tshwane University of Technology, Private Bag X680 Pretoria 0001, South Africa

---

Abstract — *With the contemporary 'softwarisation' of network resources in an SDN/NFV ecosystem, security is provided by deploying virtual security functions such as, deep packet inspection (DPI), firewall, parental control systems, intrusion detection/prevention systems (IDS/IPS) etc. However, one of the important new challenges faced by network operators is where to deploy these virtualised functions. Consequently virtual function placement solutions that simultaneously respond to the operational requirements of the network and do not compromise the security policies are needed. Hence, optimising the placement of virtual network security functions (VSNF) for various objectives is an important research issue. In this paper proposed is an On-Demand Virtual Security Services (ODVSS) model, a VSNF provisioning solution that deploys chains of virtualised security functions taking into consideration possible conflicting objectives such as load balancing, cost, coverage, traffic management, delay congestion and network security requirements. The ODVSS mathematical and metaheuristic solution model is provided. Simulation results indicate that, in contrast to other contemporary VSNF provisioning models, ODVSS significantly decreases end-to-end latency and resource utilisation for different network architectures.*

Index Terms **- Software-defined networking (SDN), network function virtualization (NFV), virtual network security functions (VSNF)**

1. INTRODUCTION

Communication Service Providers (CSP) networks today ubiquitously deploy proprietary network appliances such as firewalls, proxies, WAN optimisers, Intrusion Detection Systems (IDSs) inorder to offer various network services. However, these middleboxes are associated with high Capital Expenditures (CAPEX) and Operational Expenditures (OPEX). They are vendor specific, expensive and need trained personnel for maintenance and deployment. However with the recent softwarisation of network resources in an SDN/NFV ecosystem. It provides an alternative to the static service model for ISP and ASP [1]. The packet processing tasks are executed by software middleboxes running on commodity Servers. NFV provides opportunities for network optimisation and reduction of CAPEX and OPEX [2]. The NFV/SDN ecosystem allows for virtual security network functions (VSNFs). VSNFs can be intrusion detection systems, firewalls, deep packet inspection etc and depending on the service definition, they provide the required security services.

1.1 Network Service Request

The NFV infrastructure (NFVI), hosting the requested resources is composed of substrate nodes and link having limited available resources [3]. A service request, is composed of by a set of service function chains that interact together to provide the requested service.

There are two stages involved in resource provisioning [4]:
(i) VNF service graph composition
(ii) VNF service graph embedding

1.2 VNF service graph composition

In this stage, there is an association established between the service requests and VNF service graph links.

1.3 VNF service graph embedding

In the Second stage, the VNF-SG components (VNFs and links) are mapped into the provider's infrastructure. This involves VSNFs mapping and virtual links chaining. The mapping involves finding the best service nodes in the NFVI with enough capacity to host the VSNFs. Chaining creates optimal path that chain the VSNFs and directs traffic between them [5]. VNF chains can be required to process traffic streams with high bit rates, in order to support the real-time streaming applications that represent a large part of traffic in today's networks. Guaranteeing a given level of throughput to VNF chains is of paramount importance. In particular, the failure to provide the desired level of performance for a VNF chain may lead to poor QoE (quality of experience) for the clients [7].

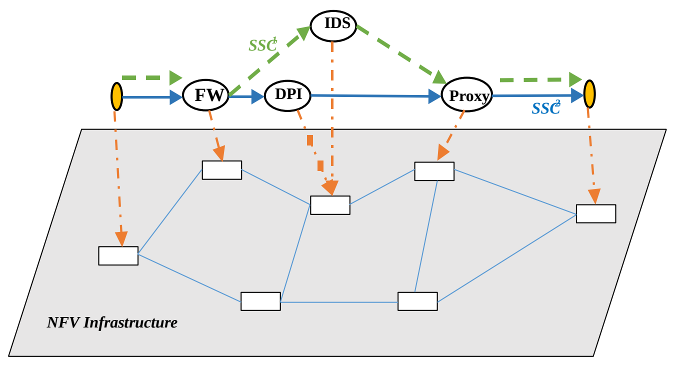


Fig. 1. A network service request as defined by ETSINFV [6]

In this paper, the efficient Virtualised Security Network Function Placement Problem (VSNF-PP) is tackled by considering performance optimisation aspects and specific security viewpoint. A novel approach to efficient provisioning of on-demand security services, On-Demand Virtual Security Services (ODVSS) model, is presented (Fig 2).

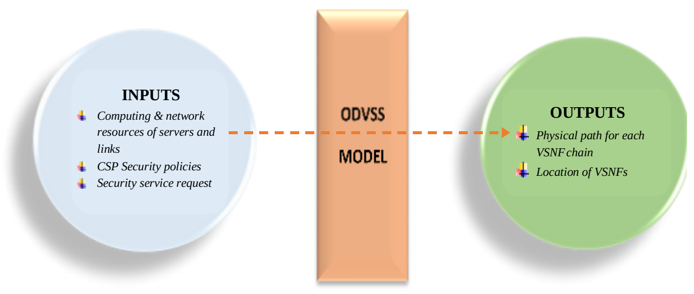


Fig. 2: ODVSS Model Workflow

Specifically, the contributions of this paper are:

- The Virtualised Security Network Function Placement Problem is formulated by the ILP model.
- A heuristic solution based on the Branch and-Cut search algorithm is proposed.

To the best of my knowledge, this work is the first attempt to address the problem of on-demand security services for accommodating mission critical and delay sensitive traffic in a network infrastructure.

1.4 Organisation of Paper

The rest of the paper is organised as follows: Section 2 reviews and discusses the related work is presented then an integer linear programming (ILP) formulation in (Section 3). Next, a meta-heuristic algorithm is proposed to obtain a near-optimal solution (Section 4). The proposed algorithm is evaluated via Matlab (Section 5). Finally, an insight for potential future work is presented (Section 6).

1.5 Motivation

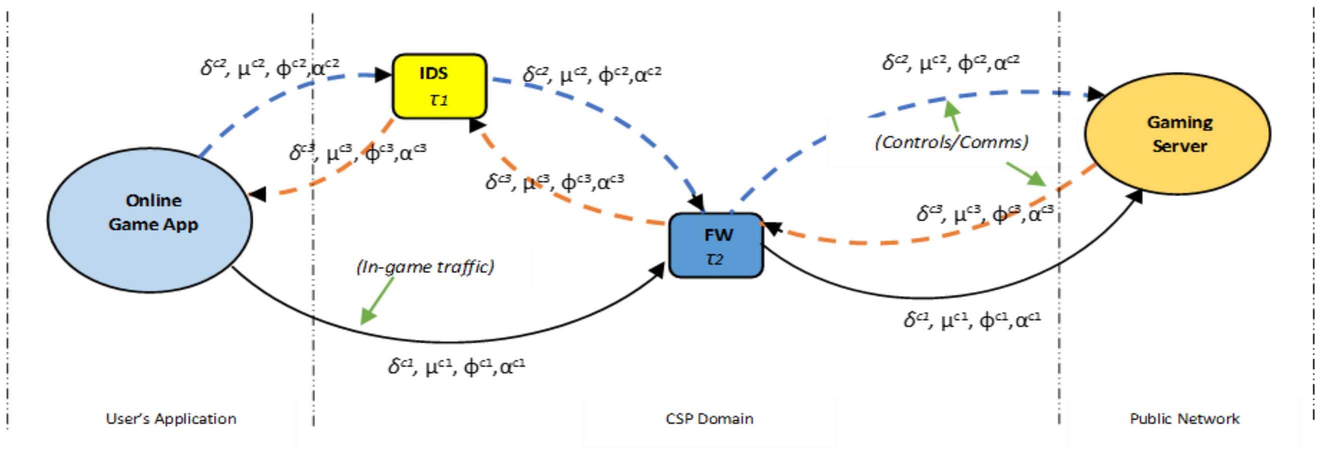
As network architecture become more complex, the number of network services also increases. Consequently the complexity leads to management challenges. In traditional networks the use of middleboxes is prevalent. Therefore the use of middleboxes is prevalent. Therefore for an additional network function another middlebox must be installed. The reduction of network complexity can be resolved by two technologies NFV and SDN.

In this paper, the Virtualised Security Network Function Placement Problem (VSNF PP) is shown to belong to the class of NP-Hard problems which are Integer Linear Programming (ILP) problems which cannot be solved in polynomial time. To circumvent this difficulty, a proposed mathematical program to answer the following questions is presented:

Q1) How to improve the flexibility of the deployment of VSNFs?
Q2) How to reduce end to end application traffic propagation delay?
Q3) How to reduce process delay in computing resources allocation for new requests?

2. RELATED WORK

The Virtualised Network Function Placement Problem (VSNFPP) in an NFV/SDN ecosystem has already attracted the research community's attention. However most of the existing work focused on service function chaining problem.
In [8] and [9], the authors proposed an ILP formulation for placing VSNFs with respect to both the security constraints. They argue that disregarding the quality of service requirements by forcing all the traffic to pass through the whole VSNFs chain can affect performance for latency-sensitive applications. The authors of [10] propose to solve the VSNFs mapping problem using a three stage model. The requested security policies are translated into chained, the security SFC mapping is formulated as a mixed ILP problem and finally, the MILP problem is solved. In [11] the authors tackled the problem of efficient Security Service Chain (SSC) deployment by proposing heuristic algorithms to choose hosting nodes and establish routing paths. The proposal avoids security service latency. In [12], the authors addressed the optimal placement of ordered sequences of VSNFs as a traveling purchaser problem. The authors in [13], improved their work by outlining network security defense patterns (NSDP) that efficiently select deployment options and capture best security practices for network security functions. In [14], the authors propose a solution that is



based on light-weight protocol. The system generate IDS dynamically in sync with traffic characteristics. In [15], the NSC embedding model presented is aimed at minimising the end-to-end latency of cross-domain chains of VSNFs. In [16], the authors propose an ILP formulation and a heuristic algorithm includes a single security related constraint to each deployed VSNF has the security level of each deployed VSNF instance is higher security level than the services request. The authors in [17], propose a model which is security deployment constraints sensitive. The proposed algorithm has the disadvantage of computing for all flows in the network and hence it not scalable. In [18], the authors used an improved version of the BFD algorithm to solve the problem of allocating VSNFs in cloud data centres. Our VSNF provisioning solution deploys chains of virtualised security functions taking into consideration possible conflicting objectives such as load balancing, cost, coverage, traffic management, delay congestion and network security requirements

3. ILP FORMULATION

In order to develop a solution to the NP hard problem, there is a proposal to address the overall problem by finding the shortest route from a source to a destination node at minimum deployment cost. This approach leads to a heuristic that will utilise Dijkstra's shortest path algorithm to solve the problem. In the following, our objective is to formulate in detail an integer linear programming (ILP) formulation.

The ILP model is constituted by three basic elements:

- ***Objective*** (goal) - that we aim to optimise
- ***Decision variables*** - that we seek to determine
- ***Constraints*** - that we need to satisfy

From the standpoint of the entire model, what we really are interested in is the optimum feasible solution. The graphical procedure includes two basic steps: (i) The determination of the solution space that defines the feasible solutions that satisfy all the constraints of the model. (ii) The determination of the optimum solution from among all the points in the feasible solutions.

3.1 Objective Formulation

In order to perform VSNF placement, our infrastructure is modelled, based on *Fig. 2*, as an edge-weighted vertex weighted directed graph denoted by $G = (N, L)$, where N represents the set of physical nodes in the nodes and L the set of the links.

The two types of nodes are:

- servers $Q \subset N$ for hosting VSNFs
- routers $T \subset N$ that are used for forwarding purposes.

Each node $n \in N$ is ascribed with a vector of available resource capacities **R** (CPU, Memory, etc.). Moreover, every link $(k,l) \in L$ is associated with a bandwidth capacity $\omega(k,l)$. Thus, the traffic demand of each SSC $f(Q^i)$ depends on to number of devices that needs to be serviced at each time. Additionally, according to the type of the VNF, the rate of the incoming traffic may differ from the rate of the outgoing traffic by a percentage $g(Q_m{}^i)$ in accordance the type of VSNFs.

Finally, regarding delay, there is an assumption that we have four communication delays:

i. Processing delay, μ_{D_p}
ii. Queuing delay, μ_{D_q}
iii. Propagation delay, $\mu_{D_{pr}}$
iv. Transmission delay, $\mu_{D_{tr}}$

3.2 Processing Delay

The processing delay $\mathrm{D_{pr}}(Q_m{}^i)$ is the time required by the node hosting the VSNF $Q_m{}^i \in Q^i$ to apply a specific operation on the packets. Therefore the processing delay id directly related to the type of VSNF and the resource capacity.

3.3 Queuing Delay
The queuing delay (n) is defined as the time the packet spends to traversing through a node. The communication devices is modelled as M/M/1 queues. Consequently minimising according to a Poisson point process Poisson point process, which has been argued as appropriate to be used for server queues [19][20].

3.4 Propagation delay
The propagation delay, D_p (u,v), is relative to the distance between two communicating nodes in a link $(u,v) \in E$.

3.5 Transmission Delay
The transmission delay is the time required for a part to transverse from a communicating device $u \in V$ to the outgoing link $(u,v) \in E$,

4. Integer Linear Programming (ILP) Formulation

The two variables are defined namely $x_i{}^s$, and $y_u{}^s{}_{,v}$. $Q_m{}^i$ is a binary variable set to 1 if the VSNF $Q_m{}^i$ is allocated on the server $u \in N$. $x_i{}^s{}_{,u}$ $y_u{}^s{}_{,v}$: the amount of traffic that flows between the Q_m^1 and $Q_{m'}^i$VSNFs that is routed over the link $(k,l) \in L$.

The goal of the *Objective Function* is as follows:
- ✓ minimise computations and resources for placing chain c
- ✓ minimise total delay experienced in chain c by minimising queuing and processing delays.
- ✓ Minimise transmission and propagation delays in links

Figure 3 above, shows a Typical Security Service Request for an Online Game Application. It shows the flow of traffic from the user application through the Communication Services Provider (CSP) domain and the public network where the gaming server is housed. Two VSNFs are shown namely, IDS (intrusion detection system) and FW (firewall). The glossary of symbols used in the IPL formulation are stated in Table I.

Objective Function:

$$\sum_{Q_m^i \in Q^i} \sum_{n \in L} [\pi_n + D_{pr}(Q_m^i) + D_q(n)] x_n{}^{Q_m^i} + \sum_{Q_m^i \in Q^i} \sum_{m' > m, Q_{m'}^i \in Q^i} \sum_{(u,v) \in L} [\varphi_{u,v} + D_p(u,v) + D_{tr}(u,v)] y_{u,v}{}^{Q_m^i Q_{m'}^i} \qquad (1)$$

Constraints
The objective function minimisation is subject to the following constraints (Equations (2)):

$$\sum \boldsymbol{P}(Q_m^i) . x_n^{Q_m^i} \leq \mathbf{O}(\mathrm{n}), \forall_n \in N \qquad (2)$$

Each node $v \in V$ is attributed with a vector of available resource capacities **O** (e.g. Memory, CPU etc.). Each edge (u, v) $\in E$ is associated with a bandwidth capacity $b(u, v)$. Each VSNFs in the Service Chain (SC) is defined by **P** (e.g Memory and CPU etc) a vector of demands (eg CPU) which needs to be fulfilled. The next constraint takes in account sources and sinks of the flows. (Equation (3)). There is need to mention that the particular constraint allows path splitting towards providing a more efficient load balancing approach:

$$\sum y_{k|l}{}^{Q_m^i Q_{m'}^i} - \sum y_{k,l}{}^{Q_m^i Q_{m'}^i} = f(Q^i) . g(Q_m^i) . \left(x_n^{Q_m^i} - x_n^{Q_m^i} \right), \forall Q_m^i, Q_{m'}^i \in Q^i, \forall_U \in V \qquad (3)$$

The constraint equation below ensures that each VSNF will be placed at only one node (Equation (4)).

$$\sum_{n \in N} x_n^{Q_m^i} = 1, \forall Q_m^i \in Q^i \qquad (4)$$

Routing Constraint (3) ensures that each node $u \in U^c$ is mapped to one physical node $i \in N$. With Constraint (4), a physical link (k,l) that can belong to a path between two i and j nodes for an arc $(u,v) \in U^c$ of chain $c \in C_Q$ only if u and v are mapped to these nodes. Constraint (5) ensures that the link (u,v) starts at one edge extending from node i to where VSNF u is mapped. Constraint (6) ensures the uniqueness of the final edges in the path. Constraint (7) is the classical flow conservation constraint. That is, an outbound flow equals an inbound flow for each intermediate node r (intermediate nodes cannot consume the flow). Constraint (8), and Constraint (9) prevents multiple links carrying traffic for a specific flow node r. Resource Constraints (10-12) ensure that the resources consumed does not exceed the available computing capacities. Security constraints ensure that the CSP's security policies are applied.

$$\sum_{i\in N} x_{i,u}^{c} = 1 \qquad \forall_c \in C_s, \forall_u \in U^c \qquad (5)$$

$$y_{u,v}^{c} \leq x_{i,u}^{c}.x_{j,v}^{c} \quad \forall_c \in C_{S,} \forall_i j \in N, \forall_{(u,v)} \in U^c, \forall_{(k,l)} \in L \quad (6)$$

$$\sum_{(i,k)\in E, j\in N} y_{i,j,u,v}^{c} . x_{i,u}^{c}. x_{j,v}^{c} = 1 \quad \forall_c \in C_{S,} \forall_{(u,v)} \in U^c \quad (7)$$

$$\sum_{(k,j)\in E, j\in N} y_{i,j,u,v}^{c} . x_{i,u}^{c}. x_{j,v}^{c} = 1 \quad \forall_c \in C_{S,} \forall_{(u,v)} \in U^c \quad (8)$$

$$\sum_{\substack{k\in N,\\(k.l)\in E}} y_{i,j,u,v}^{c} = \sum_{\substack{m\in N\\(l,m)\in E}} y_{i,j,u,v}^{c} \quad A_c \in C_s A_{i,}j \in N, \forall l \in N, l \neq i, l \neq j, \forall_{(u,v)} \in U^c \quad (9)$$

$$\sum_{\substack{k\in N,\\(k.l)\in E}} y_{i,j,u,v}^{c} \leq 1$$
$$A_c \in C_s A_{i,}j \in N, \forall l \in N, l \neq i, l \neq j, \forall_{(u,v)} \in U^c \quad (10)$$

$$\sum_{\substack{c\in C_s, i,j\in N,\\(u,v)\in U^c}} y_{i,j,u,v}^{c} . \delta^c \leq \delta'_{k,l} \forall_{(k,l)} \in L \quad (11)$$

$$\sum_{\substack{c\in C_s,\\u\in V^c}} x_{i,u}^{c} . \tau_u^c \leq \tau'_i \quad A_i \in N \quad (12)$$

4.2 Heuristic Solution

Given the complexity of the Virtualised Security Network Function Placement Problem (VSNF PP) presented in section 4, a commercial solver is incapable of producing solutions in an acceptable time. Consequently, a heuristic algorithm was implemented inorder to find near optimal solutions in shorter time.

4..2.1 Dijkstra's algorithm

The proposed heuristic will utilise Dijkstra's shortest path algorithm to solve the problem. Dijkstra's algorithm performs two kinds of operations: node selections from the candidate list Q and distance label updates. It performs a total of n node selections. Dijkstra's algorithm maintains and adjusts a vector of node distance labels {ϕ(*i*)} *i* ∈ *V* . During the computation, the set of nodes N is divided into two groups: the nodes designed as *permanently labelled* and those designed as temporarily labelled, which are stored in the candidate list Q. A permanent label associated with a node *i* represents the shortest distance from the source *s* to *i,* while a temporary label expresses an upper bound on that quantity [21].

The ODVSS Algorithm 1 can be described as follows:

Let *N* be the set of *explored nodes*.

- Let *l*(*u*) be the shortest path distance from *n* to *u*, for each *u*∈*N*.
- Initially *N* = {*n*}, and *l*(*n*) = 0.
- While *N* ≠ *V* do
- Select *v* ∉*N* with the minimum value of
- $l'(v) = \min_{(u,v), u\in N} \{l(u) + cost\ (u,v)\}$
- Add *v* to *N*, set *l*(*v*) = *l'(v).*

Initially, the source node has a permanent label equal to zero, while all other nodes a temporary label equal to ∞. The candidate list Q contains all nodes, except for the source. At each iteration, the algorithm removes from Q the node I corresponding to the minimum temporary label, designates as optimal or permanent its label and scans all outgoing edges from *i* in order to possibly update the temporary labels associated with the nodes adjacent to *i* [22].

TABLE 1: Glossary of Symbols

SET	
N	Physical nodes
L	Physical links
C_s	unidirectional chains in the service request
U^c	Virtual nodes in the chain *c*
E^c	Chain Endpoints, *c*. $E^c \subset U^c$
V^c	VSNFs in chain *c*. $V^c \subset U^c$
PARAMETERS	
λ_i	Nominal computing resources of node *i* (CPU cycles/sec)
λ_{cu}	Computing resource needed by node *u* of chain *c* (CPU cycles/sec)
$\delta_{k,l}$	Nominal capacity of link
δ_c	Minimum bandwidth required by chain c (bits/sec)
μ_{Dpr}	Propagation Delay: the time spent by a packet to traverse the link (*k,l*) (secs)
μ_{Dq}	Queuing delay: time a packet spends in chain *c* to traverse the network devices (routers and switches) in the local networks (secs)
μ_{DP}	Processing delay: the time a packet spends to traverse VSNF *u* of chain *c* placed on node *i* (secs)
μ_{Dtr}	Transmission delay: the time needed to transfer the packets from a communicating device *u* ∈ *V* to the outgoing link (*u, v*)∈ *E* (secs)
α_c	Estimated latency between the CSP network and the remote endpoint of chain *c* (secs)
$\varphi_{u,v}$	Cost for allocating a unit of bandwidth on link *(u,v)*
π_n	Allocation cost per unit of CPU on the node *i*
DECISIO N VARIABLES	
$x_{i^s,u}$	Binary variable {$x_{i}{}^{s}{}_{,u}$ = 1 if node *u* ∈ *U* is mapped to *i* ∈ *N*}
$y_{u^s,v}$	Binary variable such that $y_{u,v}^{s}$=1 if physical link (*u,v*) ∈ *L*

Dijkstra's algorithm maintains a tree *T* rooted at the source node and spanning the nodes with finite labels. *T* is stored through predecessor indices *p( )*, such that if *(i,j)* ∈*T*, then *p(j) = i*. for every tree edge *(i,j)* ∈*T*, ϕ(*j*) > ϕ(*i*) + c(*i,j*) with respect to the current distance labels ϕ(.). When all temporary labels become permanent (i.e when Q becomes empty), Dijkstra's algorithm terminates and T is a shorted path tree. The correctness can be easily proved following arguments on the cardinality of the candidate list Q [23].

Algorithm 1 ODVSS

Input: $G = (N, L)$, Security Service Request, C_s, Security Service Chains SSC
Output: $x^s_{i,u}$, $y^s_{u,v}$, $\forall_c \in C_{S,}\forall_{i,}j \in N, \forall_{(u,v)} \in U^c, \forall_{(k,l)} \in L$ (Security Service Mapping on Substrate)

```
procedure dikjstra_ODVSS
begin
Initialise decision variables $x^s_{i,u}, y^s_{u,v}$
Initialise resource capacities R (e.g., CPU, Memory)
Initialise bandwidth capacity $\omega(k,l)$.
Compute best initial solution
P:= N \ {s}; $\phi$(s) :=0; p(s) := s;  (distance label definition) $\phi(i)$: = $\infty$; p($i$) :=nil; $\forall i \in N, i \neq s$;
    while P $\neq \Theta$ do
        i= argmin $\phi(j)$: P = P \ {$i$}; (initial solution)
             j a$\in$ Q
        foreach each outgoing edge ($i,j$) do
            if $\phi(j) > \phi(i)$ + c($i,j$) then  (edge flow cost)
                $\phi(j) := \phi(i)$ + c($i,j$); p($j$) := $i$;
                Add $j$ to P if it does not already belong to P;
            end
        end
    end
Update the values of decision variables $x^s_{i,u}, y^s_{u,v}$
Update remaining capacities of constraints
Update set L If $L = \emptyset$, go to END
end procedure
```

5. EXPERIMENTAL SETUP AND RESULTS

5.1 Methodology

In this experiment, a comparison of the ODVSS ILP and heuristic on random topologies with 30 nodes and 48 links is made. A CSP configures a security service request for user applications. For this evaluation requests are generated automatically and they have random number of chains. Initially, simulation of service requests is undertaking using the ODVSS heuristic. The solver is run to compute an optimal solution for individual requests.

A security service request is configured by the TSP to provision security for user applications. For evaluation purposes, we automatically generate requests composed of a random number of chains. The security service requests are generated using a Poisson process with exponential distribution of inter-arrival and holding times. Once a service expires, the resources allocated to it are released. We start by simulating the processing of service requests using the ODVSS heuristic.

5.2 Simulation

The simulations are carried out on an Intel(R) Core (TM) i5-8250U CPU @ 1.80GHz and 8 GB RAM workstation and we use MATLAB 2013a with the CLP toolbox to solve the ILP.

5.3 Evaluation

We then prove the benefits of the proposed ODVSS approach is compared with other baseline heuristics namely Breadth First Search (BFS) [24], Tabu Search (TS) [25] and Greedy Fast Processing (GFP) [26] which have already been studied in literature. The ODVSS heuristic places all the VSNFs on a single path to efficiently address that Constraint (1-12). The heuristic places the VSNFs chains on nodes in the chosen path.

5.3 Discussion

Chart 1 compares the performance of the ODVSS provisioning algorithm and the baseline approaches BFS, TS and GFP on random networks (30 nodes and 48 links).

(i) Consumed CPU resources:

A comparison is made between consumed resources and load. What is the average number of security service requests.

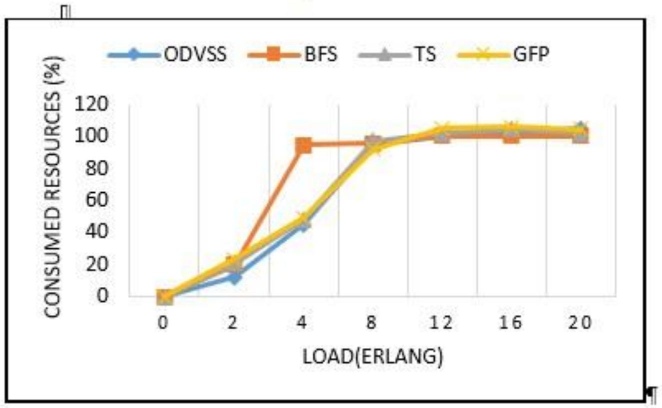


Chart 1: Consumed CPU Resources

(ii) Length of service chain:
The adaption of the comparison schemes to different lengths of service chains is analysed. Four ranges of lengths are tested namely: [1,2], [3,4], [5,6], and [7,8]. The results are shown in Chart 2. Since a longer chain consumes more resources, the total size of accepted demands decreases as the length of the service chains increases. The ODVSS performs well by taking the shortest path to utilise the precious communication resource efficiently. Our solution outperforms all the heuristics, especially for the more challenging cases, i.e., demands with longer service chains. This from the use of dynamic programming to maximise the reuse factor of a path.

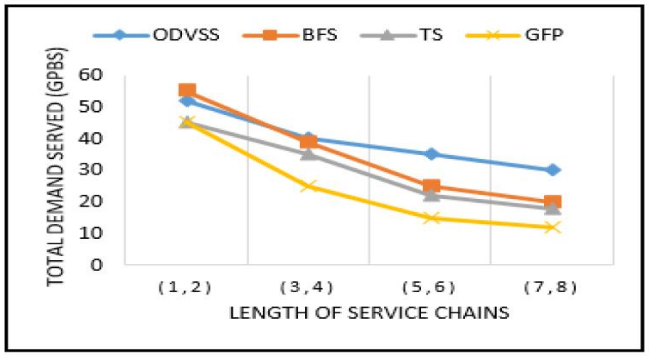


Chart 2: Length of Service Chains

(iii) Processing Delay: Chart 3 shows that the ODVSS decreases end-to-end latency and increase the number of active services in the network. Consequently, ODVSS performs better because the least loaded nodes are likely to have shorter queues, and services mapped to the node are processed quickly.

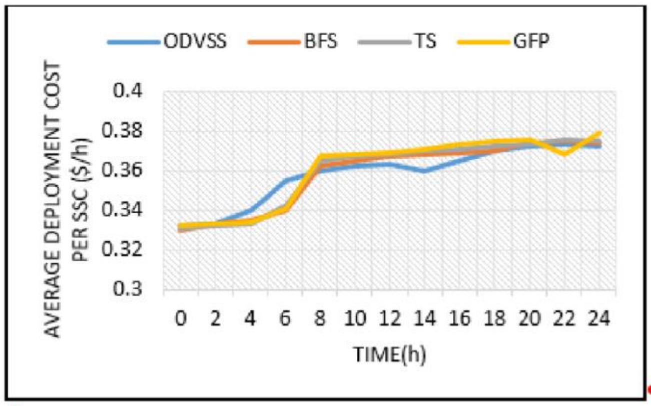


Chart 3: Average Deployment Cost per Security Service Chain

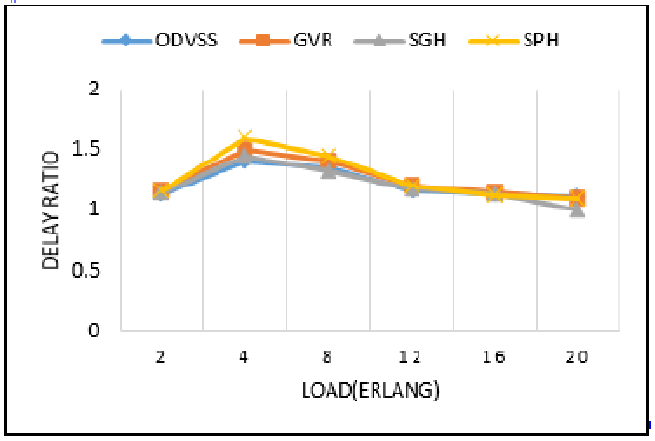


Chart 4: Delay Ratio

As evidenced by the results shown in Chart 4**,** ODVSS performs slightly better than the other heuristic algorithms because of its ability to iteratively improve the solution,

6. CONCLUSION

This paper focuses on the security aspects in an SDNFV ecosystem with the intention of answering the following questions: how to improve the flexibility of the deployment of VSNFs, how to reduce end to end application traffic propagation delay and how to reduce process delay. The complexity of the Virtual Security Network Function Placement Problem by proposing an ODVSS approach was addressed. The model is a unique VSNF provisioning solution that deploys chains of virtualised security functions taking cognisance of the need to attend to possible conflicting objectives of the network and user applications security requirements. The underlying principle behind our design and provided the ODVSS mathematical and heuristic solution model was discussed. Simulation results indicate that, in contrast to other contemporary VSNF provisioning models, ODVSS significantly decreases resource utilisation and end-to-end latency for different network topologies. In future work, it is proposed to validate the algorithm for larger networks using complex optimisation packages.

REFERENCES

[1] M. M. Tajiki et al., “Joint Energy Efficient and QoSaware Path Allocation and VNF Placement for Service Function Chaining,” IEEE TNSM, (2018).
[2] NFV/SDN combination framework for provisioning and managing virtual firewalls", IEEE Conference on Network Function Virtualization and Software Defined Network, San Francisco, CA, USA, 107–114, 2016.
[3] A. Ali, C. Anagnostopoulos, and D. P. Pezaros, “Resource aware placement of softwarised security services in cloud data centers,” in 2017 13th

International Conference on Network and Service Management (CNSM), Nov 2017, pp. 1 – 5.

[4] Y. Liu, H. Zhang, J. Liu, and Y. Yang, "A new approach for delivering customized security everywhere: Security service chain," in Security and Communication Networks, 2017.

[5] T. Lin, Z. Zhou, M. Tornatore, and B. Mukherjee, "Demandaware network function placement," J. Lightw. Technol., vol. 34, no. 11, pp. 2590–2600, Jun. 1, 2016.

[6] R. Doriguzzi-Corin, S. Scott-Hayward, D. Siracusa, and E. Salvadori, "Application-centric provisioning of virtual security network functions," in 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFVSDN), Nov 2017, pp. 276–279.

[7] I. Jang, S. Choo, M. Kim, S. Pack, and M. K. Shin, "Optimal network resource utilization in service function chaining," in Proc. IEEE NetSoft Conf. Workshops (NetSoft), Jun. 2016, pp. 11–16.

[8] W. Ma, B. Jonathan, Z. Pan, D. Pan, N. Pissinou, "SDN-Based Traffic Aware Placement of NFV Middleboxes", IEEE Transactions on Network and Service Management, 14:528– 542, (2017). .

[9] X. Wang, C. Wu, F. Le, A. Liu, Z. Li, F. Lau, "Online VNF scaling in datacenters", IEEE 9th International Conference on Cloud Computing, San Francisco, CA, USA, 140–147, (2017).

[10] A. Alleg, R. Kouah, S. Moussaoui, T. Ahmed, "Virtual Network Functions Placement and Chaining for real-time applications," IEEE 22th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, Lund, Sweden, 1-6, (2017).

[11] A. Shameli-Sendi, Y. Jarraya, M. Fekih-Ahmed, M. Pourzandi, C. Talhi, M. Cheriet, "Optimal placement of sequentially ordered virtual security appliances in the cloud", IFIP/IEEE International Symposium on Integrated Network Management, Ottawa, ON, Canada, 818 –821, 2015

[12] P. Murukan, D. Jamaluddine,"A Cost-based Placement Algorithm for Multiple Virtual Security Appliances in Cloud using SDN: MO-UFLP (Multi-Ordered Uncapacitated Facility Location Problem)", http://arxiv.org/abs/ 1602.08155, (2016).

[13] X. Wang, C. Wu, F. Le, A. Liu, Z. Li, F. Lau, "Online VNF scaling in datacenters", IEEE 9th International Conference on Cloud Computing, San Francisco, CA, USA, (2017).

[14] T.W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," in Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM), Apr. 2016, pp. 1 – 9.

[15] D. Cho, J. Taheri, A.Y. Zomaya, P. Bouvry, P. "Real-Time Virtual Network Function (VNF) Migration toward Low Network Latency in Cloud Environments", IEEE 10th International Conference on Cloud Computing, Honolulu, CA, USA, 798–801, (2017).

[16] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," IEEE Trans. Netw. Service Manage., vol. 13, no. 3, pp. 518–532, Sep. 2016.

[17] T. Lin, Z. Zhou, M. Tornatore, and B. Mukherjee, "Demandaware network function placement," J. Lightw. Technol., vol. 34, no. 11, pp. 2590–2600, Jun. 1, 2016.

[18] I. Jang, S. Choo, M. Kim, S. Pack, and M. K. Shin, "Optimal network resource utilization in service function chaining," in Proc. IEEE NetSoft Conf. Workshops (NetSoft), Jun. 2016, pp.11–16.

[19] A. Leivadeas, M. Falkner, I. Lambadaris, and G. Kesidis, "Optimal virtualized network function allocation for an SDN enabled cloud," Comput. Standard Interfaces, vol. 54, no. 4, pp. 266–278, 2017.

[20] M. Falkner, A. Leivadeas, I. Lambadaris, and G. Kesidis, "Performance analysis of virtualized network functions on virtualized systems architectures," in Proc. IEEE CAMAD, Mar. 2016, pp. 1 – 9.

[21] R. Doriguzzi-Corin, S. Scott-Hayward, D. Siracusa, M. Savi, E. Salvadori, "Dynamic and Application-Aware Provisioning of Chained Virtual Security Network Functions", https://arxiv.org/pdf/ 1901.01704.pdf, (2019).

[22] S. Demirci, M. Demirci, S. Sagiroglu, "Optimal Placement of Virtual Security Functions to Minimize Energy Consumption", International Symposium on Networks, Computers, and Communications, Rome, Italy, 1–6, (2018).

[23] A. Leivadeas, M. Falkner, I. Lambadaris, and G. Kesidis, "Optimal virtualized network function allocation for an SDN enabled cloud," Comput. Standard Interfaces, vol. 54, no. 4, pp. 266 –278, 2017.

[24] H. Hu, G. Ahn, "Virtualizing and Utilizing Network Security Functions for Securing Software Defined Infrastructure", NSF Workshop on Software Defined Infrastructures and Software Defined Exchanges, Washington, D.C., USA, 70, (2016).

[25] Y. Liu, H.-Q. Zhang, J. Liu, and Y.-J. Yang, ''A new approach for delivering customized security everywhere: Security service chain,'' Secur. Commun. Netw., vol. 2017, no. 3, 2017, Art. no. 9534754.

[26] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, ''Design and evaluation of algorithms for mapping and scheduling of virtual network functions,'' in Proc. 1st IEEE Conf. Netw. Softwarization, Apr. 2015.

[27] L. Yicen, Yu Lu, Wenxin Qiao, And Xingkai Chen, A "Dynamic Composition Mechanism of Security Service Chaining Oriented to SDN/NFV-Enabled Networks", IEEE Access, (2018).

BIOGRAPHY

Godwin T.S. Chapanduka is currently studying for a Phd in Electrical Engineering at Tshwane University of Technology (TUT), Pretoria, South Africa. His research interests include 5G and Beyond Networks, network security, Internet of Things, design and performance evaluation of the next generation of wireless network architecture, high-speed networking, SDN, and NFV.