In-Network Placement of Security VNFs in Multi-Tenant Data Centers

Abeer Ali[§]
School of Computing Science
University of Glasgow, UK
A.Ali.4@research.gla.ac.uk

Christos Anagnostopoulos
School of Computing Science
University of Glasgow, UK
Christos.Anagnostopoulos@glasgow.ac.uk

Dimitrios P. Pezaros
School of Computing Science
University of Glasgow, UK
Dimitrios.Pezaros@glasgow.ac.uk

***Abstract*—Middleboxes are typically hardware-accelerated appliances such as firewalls, Proxies, WAN optimizers, and NATs that play an important role in service provisioning over today's Data Centers. We focus on the placement of virtualised security services in multi-tenant Data Centers. Customised security services are provided to tenants as software VNF modules collocated with switches in the network. Our placement formulation satisfies the allocation constraints while maintaining efficient management of the infrastructure resources. We propose a Constraint Programming (CP) formulation and a CPLEX implementation. We also formulate a heuristic-based algorithm to solve larger instances of the placement problem. Extensive evaluation of the algorithms has been conducted, demonstrating that the VNF approach provides more than 50% reduction in resource consumption compared to other heuristic algorithms.**

***Index Terms*—Data Centers security, resource management, security network functions, VNF orchestration**

I. Introduction

To mitigate the problems of the legacy hardware-based middleboxes such as, e.g., expensiveness, vendor lock-in, deployment inflexibility, and lack of resource scalability [1], [2], virtualised middleboxes have emerged such as WAN optimizers [3] and IDPS systems [4], [5] while the Network Function Virtualization (NFV) technology have been used to deploy virtualised middleboxes as network functions (NFs).

Contribution: In this paper, we contribute by developing efficient allocation for security services over multi-tenant virtualised DCs. A pool of security services such as firewalls, IDS/IPS, DDoS mitigation tools, and DPI engines, are available for tenants to request in the form of software modules. The modules requested by each tenant will be deployed in the Data Center (DC) as VNFs where they can process traffic destined to the requesting tenant. We then address the problem of allocation of different security modules over a virtualised DC infrastructure using a *resource-aware placement methodology*. Our contributions include introducing the placement problem for both stateless and stateful class of the security function. We formalise a Constraint Programming (CP) model of the security function placement as an instance of the NP-hard bin packing optimisation problem with an objective function based on maximising residual resources. For a polynomial-time solution, a modified version of the Best-Fit Decreasing (BFD) greedy algorithm is also proposed. Finally, through a comparative assessment, we demonstrate that BFD optimises the residual resources when compared to other algorithms. The remainder of this paper is structured as follows: Section II discusses related work, and Section III introduces the problem of security modules placement. Section IV formalises the problem in CP form and, in Section V, we adopt a modified version of the BFD algorithm and provide a performance and comparative assessment against different algorithms in Section VI. Finally, Section VII concludes the paper.

[§]Electronic Research Institute (ERI), Egypt

II. Related Work

For virtualised middleboxes, research is focusing on the automatic managing of network functions such as VNFs which known as VNF orchestration problem. The earlier work focused on particular aspects of an overarching architecture. For example OpenNF in [6] is a centralised control plane that orchestrates NF dynamically. It supports scaling through duplication and proposes coping the internal states and network forwarding states instead of copying the whole virtual machine to reduce migration cost. While the problem of VNF placement is an important part of any orchestration tool, approaching this problem highly depends on the considered parameters such as computing cost, link bandwidth, QoS , economical profit, network load, energy efficiency, security.. etc. In [7], authors target reducing the energy consumption using a consolidation algorithm based on a migration policy of chained VNFs. In [8], the VNF placement problem is formulated as a series of scheduling decision aims at minimising the latency of VNF scheduling by assigning the execution time slots to different services traversing the same VNF.

However, the distinct constraints related to security functions such as, the granularity of the processed traffic are neglected by most studies of the automatic management of VNFs [9]. We explicitly address the requirements and constraints of security services which distinguish our work from previous research in the management of virtualised services, e.g., our work addresses the traffic constraints of security functions which impose a restriction on allocation. We propose the deployment of the security modules in-network and on traffic path to reduces the overhead of having to resort to non-shortest-path routing while most previous work proposes to place NFs on DC servers [10], [11]. Furthermore, specific security functionality that is based on building a behavioural

model for traffic such as, e.g., anomaly detection modules, cannot be shared since each tenant will have a different normal behavioural model. Therefore, work reduces the complexity and security risks of deploying shared modules by adopting the non-sharing strategy.

III. Multi-tenant Security Management

In multi-tenant virtualised DCs, customised security is essential where each tenant run different services; each service requires a different security level. For instance, a typical web server may require security modules to detect and mitigate HTTP flood attacks and SQL injections while critical servers may require a firewall, IDS and/or DPI to guarantee high availability and data integrity. We focus on orchestrating security services in multi-tenant virtualised DCs where they are offered as software modules that are allocated throughout the infrastructure as VNFs to process the required traffic. The services are offered on a per-tenant basis, and each request will result in deploying a function to process traffic of the requesting tenant.

A. Fat-Tree Architecture

A multi-rooted tree is one of the most common virtualised DC network architectures [12]. For example, a k=4 fat-tree topology shown in Figure 1 with three layers of switches (ToR, aggregation, and core). We assume routing is flow-based Equal Cost Multiple Path (ECMP), where flows are distributed over equal cost links. Security modules are allocated to points collocated with switches at all layers. Traffic is rerouted from switches to the security function and back as shown in Figure 1. This approach will reduce the detour length of the path that traffic has to take to pass through a security function. The security function abstraction can be implemented as a distinct namespace within a software switch or on a separate, virtualised commodity x86 architecture that physically connects to a traffic-forwarding switch.

B. Placement Strategy

For security function, traffic and resource constraints must be considered in the process of selecting where the function is going to be allocated in the infrastructure. The placement strategy involves satisfies the security service request of a tenant by selecting an allocation for the security functions that provide the security service. Security functions can be classified into two classes based on how traffic is processed to detect threats and subsequently the level of aggregation of traffic packets required for each. This classification will specify the traffic constraints of the modules and guide the placement algorithm to select the location where a security function can accurately capture the required level of traffic aggregation [9].

The **Stateless class** modules process traffic at the individual flow or packet level. For example, a module of this class has a set of signatures database or access list, and it operates by finding a match when compared their databases with the patterns of the packet or flow of the incoming traffic. The matching process depends on the state of a single packet or flow. Therefore, it can operate independently at different links in case of per-flow routing. Consequently, VNF of this class can be duplicated across network locations where tenant's traffic is being split, as long as the routing is per-flow. Examples of this equivalence class are firewalls and signature-based IDS.

The **Stateful class** modules process traffic to detect anomalies based on a coarser traffic granularity such as,e.g., flow-aggregation on the contradictory to the stateless class which process traffic on fine granularity such as single packet or flow. The stateful class modules depend on traffic features such as volume where changes in traffic volume can indicate anomalies. While others depend on deviations in traffic distribution features such as Entropy or Histograms [13]. More complicated modules will use statistical learning techniques. The modules of this class start by constructing a normal behaviour model to the traffic under inspection by extracting information from flow aggregate features then under operation it will detect anomalies based on deviations of current processed traffic from the stored normal model. However, capturing an accurate behaviour model will require all the flows of the monitored traffic to be rerouted to one instance of the network function of this type, and no replication can be allowed in case of traffic distribution on multiple links.

This approach has been widely adopted and involves all the data continuously pushed to a central point for analysis [14], others propose that computational, transmission and storage cost of monitoring traffic for anomaly detection can be reduced by mapping the traffic to less dimensional space by extracting features and only transmit these features in case of distributed nodes monitoring [15]–[18]. We adopt the latter approach in allocating functions of the stateful security class, instead of steering traffic to one instance, monitoring nodes will be deployed wherever traffic is distributed. These nodes will extract the needed features from the traffic and share this information with the main node that takes the detection decision.

The resource constraint of a security function will guarantee that the required resources for the function operation are available. Each available locations has a predefined capacity of resources. Each module in the system pool will have two vectors that represent the required resources. The first vector is the *baseline requirement* represent the resources required for initial deployment of the module and the second vector is the *traffic requirement* which represents the resource required to process a unit of the traffic.

IV. Constraint Programming Approach

A. Problem Description

The placement of the virtualised functions of security modules to satisfy the security services requested by tenants is a resource allocation problem. It allocates the security functions in distributed locations in virtualised DC which each location has a limited capacity. We address the static placement of the two security classes introduced in Section III. The

placement will ensure the traffic constraint is satisfied in case of allocations where traffic is distributed over multiple links.

For example, in the k=4 fat-tree shown in Figure 1, a stateless class module deployed for a tenant in the server in red has three available locations, the first allocation is where one instance will be allocated to the TOR switch. Second allocation, where two instances will be allocated to aggregation level switches (shown in red). Third allocation, where four instances will be allocated to all core level switches. For the stateful class, ToR level is the same as the stateless one, while for aggregation level allocation deployed for a tenant on the server shown in green, the main instance will be deployed in one switch and a monitoring instance will be deployed on the other switches (shown in green in Figure 1). There will be no steering of traffic, however, shared data will be directed from the monitoring instance to the main instance shown by the orange links. The same applies for the core level where the main instance will be deployed in one of the four core switches and monitoring instances in the other three with communicate with the main instance to share information. Moreover, the proposed placement strategy can be applied to any network architecture as it consider traffic split on links due to ECMP routing and .

When more than one allocations satisfy the resource and traffic constraints of a request, the one that optimises the placement objective will be selected. We design the placement algorithm to optimise the resources usage. We accomplish this with two objectives, the first objective is to maximise the *residual resources* of the framework, which represents the spare computing resources of the locations after the placement has been completed. The second objective is to minimise the *communication overhead* of the allocation which represents the communication bandwidth overhead due to the allocation and targeting minimise bandwidth usage for the placement.

B. Problem Formulation

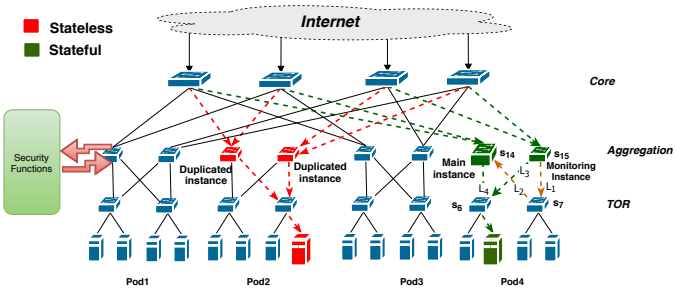


Fig. 1. Fat Tree Cloud Data Center; Size k=4

We represent the placement problem of security functions in virtualised DCs as an instance of a variable cost – variable size bin packing problem (VSBPP) [9], [19]. Therefore, every request will be associated with a tenant and a security function requested by that tenant. Every location available for allocation has two dimensions of resources, switches dimension and links dimension. The switches dimension will represent the required resources of the set of switches for a request to be allocated relative to function required resources, while the links dimension will be the required bandwidth capacity of the links relative to the function communication overhead to allocate the request in the switches. Locations that represent allocating requests in TOR switches can be used to allocate both types of security functions and has empty links dimension, while locations represent higher layers allocations are designed for only one type. Therefore, every request can be only associated with locations that could allocate its function type. To minimise cost, allocation will reduce duplicates and subsequently keeping allocation in the lower levels of the hierarchy while maintaining resources and traffic constraint of allocating the functions satisfied.

The constraint programming (CP) formulation of problem is as follows: Let the overall framework include a set of $q > 0$ requests $Q=\{r_1, r_2, ...r_q\}$ with each request r representing a function requested by a tenant each with attribute $r.type$ that represents the type of security function requested in r, $r.type$ can be $stateless$ or $stateful$, set of $m > 0$ switches $S=\{s_1, s_2, ..., s_m\}$ each with attribute $s.c$ as the resources capacity of s, a set of $n > 0$ links $L=\{l_1, l_2, ...l_n\}$ each with attribute $l.b$ as the bandwidth capacity of link l and attribute $l.w$ as the link weight, a set of k locations $P=\{p_1, p_2, ...p_k\}$, each with attribute $p.type$ that represent the type(s) of security functions that be allocated in p, switches and links dimensions for locations for different request presented in the following two metrics. A matrix u of size k x q x m representing the switches resources cost of allocation where $u_{p,r,s}$ is location p required resources to be allocated in switch s to satisfy request r; or zero if location p can not accommodate request r due to function type or does not require any resources to be allocated in switch s. A matrix v of size k x q x n representing the communication cost of allocations where $v_{p,r,l}$ is location p required bandwidth to be allocated in link l to satisfy request r; or zero if location p can not accommodate request r due to function type or does not require any bandwidth to allocated in link l. We have two objectives: the first objective is to maximise the *residual resources* (RS) which is represented the spare resources in switches after placement. The second objective is to minimise *Communication Overhead* (CO) which the communication cost of the placement that is represented as the sum of communication overhead traffic rate on each link multiplied by the link weight.

The allocation is represented as a binary variable x of size pxk where $x_{r,p}$=1 when request r is allocated to location p; 0 otherwise. The CP formulation is as follows:

$$\max . \Big[\sum_{\forall s \in S} s.c - \sum_{\forall r \in Q} \sum_{\forall p \in P} \sum_{\forall s \in S} x_{r,p} \cdot u_{p,r,s} \Big] \quad (1)$$

$$\min . \Big[\sum_{\forall r \in Q} \sum_{\forall p \in P} \sum_{\forall l \in L} x_{r,p} \cdot v_{p,r,l} \cdot l.w \Big] \quad (2)$$

$$\text{s.t.} \sum_{\forall r \in Q} \sum_{\forall p \in P} x_{r,p} \cdot u_{p,r,s} \leq s.c \quad \forall s \in S \quad (3)$$

$$\sum_{\forall r \in Q} \sum_{\forall p \in P} x_{r,p} \cdot v_{p,r,l} \ \leq l.b \quad \forall l \in L \tag{4}$$

$$x_{r,p} = 0, \quad \forall r \in Q, \ \forall p \in P \qquad if \ r.type \notin p.type \tag{5}$$

$$\sum_{\forall p \in P} x_{r,p} = 1, \quad \forall r \in Q \tag{6}$$

Note: the first objective function of the CP formulation is represented in (1) as maximising the residual resources after the placement. The Second objective function of the CP formulation is represented in (2) as minimising the communication overhead after the placement. The constraint in (3) represents the switches capacity constraints. The constraint in (4) represents the links capacity constraints. The constraint in (5) represents the location-validity for requests where the function type of the request must be satisfied by the location. The constraint in (6) ensures that each request is allocated to one location.

Two versions of the Constraint Programming (CP) model was considered, the first version called CP_RS+CO which optimise for the sum of both function with equal weights for each as stated in the following function and with the objective in equation 1 converted to minimise consumed resources instead of maximising residual resources. This method gives both functions equal weight in the optimisation. The second version called CP Two-pass which consists of a two-pass method optimisation. First, it optimises the placement of the stateful requests for the objective in equation 2 and then it optimises for the objective in equation 1 against the stateless requests after encoding the optimal solution obtained in the first optimisation as a hard constraint. This method optimises the objective in equation 2 as a primary goal to the problem, while the objective in equation 1 is considered a secondary goal.

V. Methodology

Heuristic algorithms are proposed as a solution to the NP-hard bin-packing problems [20]. The widely used Best Fit Decreasing (BFD) algorithm can solve the problem within polynomial time [21]. We have adopted a modified version of BFD to allocate virtual machines (VM) to locations that result in the least increase in power consumption [19]. We previously proposed a computing resources version of that algorithm to solve the stateless function allocation problem [9] . In the resource-aware BFD algorithm, Q is a set of initial requests, each represent a tenant's request to certain module, L is a set of locations available for allocating requests and the set A refers to the set of allocated requests in certain locations, the function $sort(Q)$ sorts the requests from Q by a decreasing order of baseline resources required; $capacity(A, r, l)$ ensures the resources required in location l in allocation A is enough to accommodate a give request r; $validation(r, l)$ constrains the location to those who satisfy the traffic constraints such as for a stateful class request r, only locations that associated with communication links are valid, while $total\_cost(r, l)$ calculates the total cost of allocating the request r to location l which represented of computing cost plus communication cost if any in the contrary to the work in [9] which only consider computing resources cost, as illustrated by the following algorithm

Input: Set of requests Q, set of locations L
Output: Set of requests allocated to locations A

$A \leftarrow \emptyset$ // initialisation
$Q^* \leftarrow sort(Q)$ // sort request w.r.t. resources
for all $r \in Q^*$ **do**
  for all $l \in L$ **do**
    if $(capacity(A, r, l)$=TRUE$) \bigwedge (validation(r, l)$=TRUE$)$ **then**
      $l^* = \arg\min_{l' \in L} total\_cost(r, l')$
    end if
    if $(l^* \neq 0)$ **then**
      $A \leftarrow A \cup \{(r, l^*)\}$ // allocate request r to location l^*
    end if
  end for
end for
return Set of allocated requests A

VI. Performance Evaluation

A. Performance metrics

To demonstrate the efficiency of the resource allocation algorithms, we introduce two metrics that represent our objectives presented in Section IV. The first metric is the *Residual Resources* (RS) of the network, which is the ratio of the spare resources (after placement) to the total amount of resources available and is calculated by adding the residual resources at each location after placement. RS is a normalisation of our first objective presented in Section IV. The second metric is *Communication Overhead* (CO) of the allocation as the total of the communication overhead resulting from sharing information by the stateful class which represent the sum of communication overhead traffic rate on each link multiplied by the link weight. CO is a normalisation by the total consumed bandwidth to represent our second objective presented in Section IV.

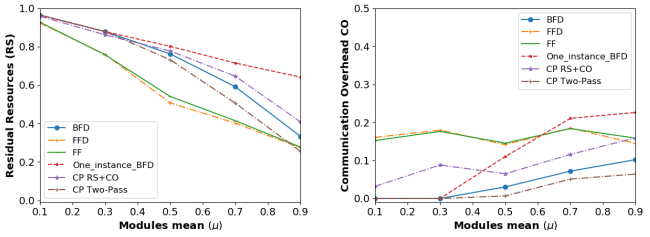
B. Models Comparison

We compare the proposed BFD algorithm two resource allocation algorithms: First Fit (FF), and First Fit Decreasing (FFD) [22]. Specifically, in FF, the unordered requests are allocated to the first level that will fit them. In FFD, the requests are ordered in decreasing order based on resource consumption and are allocated to first fit (module types with high resource consumption allocated first). In BFD, the requests are ordered the same way as in FFD, and then they are allocated to the best-fit location where the total cost is minimised. In addition, two variants of constraint programming solutions were modelled using the CPLEX optimizer [23] and then used as a baseline for comparison with other solutions. The two modelled are the RS+CO model and the two-pass model previously discussed in IV. Furthermore, we compare the proposed BFD algorithm with *one_instance_BFD* of the algorithm. The one_instance version simulates the legacy allocation of hardware middleboxes where in case of traffic distributed over multiple links, all traffic must are steered to one instance of the module.

C. Performance Assessment

Without loss of generality, we assume traffic is uniformly distributed on all servers and each server represents one tenant. To test our allocation strategy overhead on the network structure on all levels, we assign switches an initial capacity that enough to accommodate the same number of requested in each level. Because duplication will increase allocation cost in higher levels, switches initial capacity will increase based on duplication cost expected at each level. We simulated six different security function. A security function will have a probability p to be stateless and $(1-p)$ to be stateful. The resources requested and the communication overhead for each module are drawn for a normal distribution. The tenant request rate is the number of functions requested by a tenant.

We simulated the increase of *workload* over the network in two dimensions the modules required resources and the traffic demand of tenant. The former represented as the mean of the normal distributed which resources requirements of the modules are drawn from. The later represented as the mean of the normal distributed which traffic demand rate of tenants is drawn from. We consider resources as a one-dimensional vector. All results are computed over an average of 10 runs. We present results of the RS and CO for the following experiments.

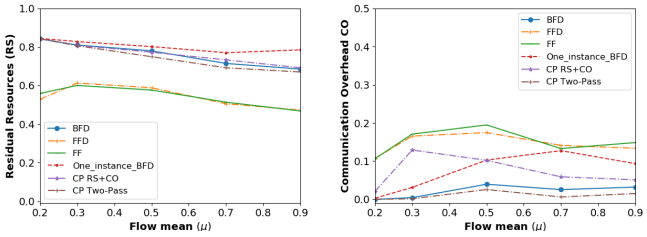


(a) Residual Resources (b) Communication Overhead

Fig. 2. Residual Resources and Communication Overhead of BFD, FFD, FF, and CP algorithms when k=6

The first experiment shows the *effect of the modules sizes workload* on the performance metrics in case of traffic demand rate parameters are μ=50% and σ=10% of maximum value of flow rate, tenant request rate equal 1, communication overhead parameters are μ=50% and σ=10% of max value of flow rate between shared nodes, base_part to traffic_part percentage is 50%. The results of RS and CO metrics for a k=6 Fat-tree are shown in Figure 2. Figure 2(a) shows the *objective function* Residual Resources (RS) starts decreasing linearly with the workload, where the increase of requested resources will result in a reduction in spare resources. While all algorithms suffer from such reduction, Best-fit Decreasing BFD shows less reduction in RS than (FFD and FF). This can be attributed to Best-fit algorithms utilise resources by selecting locations which cost the least increase in resource consumption, allowing more resources to the allocation process, and leading to an increase in RS that can reach 50% of other algorithms. Additionally, Figure 2(a) shows the middleboxes one instance legacy allocation strategy where requests are allocated to one instance of requested function and all traffic must be steered to that instance. For comparison reasons, we adapt the BFD version of the one_instance algorithm. The results show that one_instance_BFD algorithm has more spare resources than other algorithms which are a result of only one instance are deployed for each request and no duplication is endured. CP RS+CO are showing more RS than CP two-pass because it balances between the two objectives while CP two-pass gives priority to CO over RS, however, CP RS+CO only show 10% more RS than BFD.

Figure 2(b) shows the Communication Overhead (CO) *objective function* after the allocation is complete and, the results show that at low workload, less than 0.3, best-fit algorithms (BFD and one_inctance_BFD) show no communication overhead while FFD and FF show increased overhead. This is a result of the FFD and FF allocate requests in random locations which will result in allocations in higher layers of the network which cause a communication overhead while Best-fit algorithm allocates requests to less resource consuming location first which are TOR switches which have no communication overhead. In higher workload, CO increases with modules required resources which indicate that more requests are allocated to higher layers which impose a communication overhead of shared information between main and monitoring instances of the stateful class. However, BFD algorithm still shows less communication overhead than other algorithms due to the best-allocation strategy that can reduce communication overhead up to 80% of the consumed bandwidth. Opposite to RS one_instance_BFD show, there is a linear increase in CO as traffic steered to the centralised instance will cause a significant communication overhead compared to the distributed allocation strategy adopted by other algorithms while CP Two-pass shows the least communication overhead compared to other algorithms where is priorities CO over RS functions. The Figure also shows that BFD has less communication overhead than CP RS+CO. However, it shows less than 10% more communication overhead than CP Two-pass.



(a) Residual Resources (b) Communication Overhead

Fig. 3. Residual Resources and Communication Overhead of BFD, FFD, FF and CP algorithms when k=6

The second experiment shows the *effect of traffic demand of tenant as workload* on the performance metrics in case of modules size/required_resources parameters are μ=50% and σ=10% of the maximum value of module size and the same

parameters for the previous experiment. The results of RS and CO metrics for a k=6 Fat-tree are shown in Figure 3. Figure 3(a) shows Residual Resources slightly decreases with the workload, where the increase of traffic rate results in increasing of required resources as the traffic part of functions' required resources is depending on traffic rate and results in a reduction in spare resources. While all algorithms suffer from such reduction, Best-fit Decreasing algorithms BFD still show less reduction than (FFD and FF) that reach up to more than 20% in spare resources. Similar to Figure 2(a), Figure 3(a) shows that one_instance_BFD algorithm has more spare resources than other algorithms which are a result of one instance per request are deployed. Furthermore, CP RS+CO and CP Two-pass show the same trend as RS is slightly affected by increasing network flow while they and BFD still showing a significant higher RS compared to FF and FFD. In Figure 3(b), the results show that CO shows steady results for the BFD algorithm due to that CO is normalised to the total consumed bandwidth which also increases with the workload. On the other hand, FFD and FF algorithms show a reduction in CO with workload increase and this can be attributed to as all locations start to fill up, FFD and FF will allocate more requests to lower layers of the network which impose no communication overhead to the system. The same as Figure 2(b), BFD shows less CO than (FFD, FF and one_instance_BFD) that reach up to 80%. The figure also shows that CP two-pass has the least communication overhead compared to other types including the CP_RS+CO due to it prioritise CO over RS.

Based on the above results, the placement strategy of in-network placement of security function with the adoption of sharing information between instances presented in BFD algorithm was able to minimise the communication overhead. However, the computing resources have increased due to the duplication strategy compared to the non-duplication algorithm in a trade-off between computing resources and communication overhead. Additionally, the BFD showed optimisation for our two objective functions compared to other heuristic algorithms and the Legacy one_instance allocation adopted from middleboxes.

VII. Conclusions

In this paper, we have studied the problem of In-Network placement of security VNFs over multi-tenant Data Centers. We have classified security functions based on the allocation constraints as traffic constraints and computational resource requirements. We have formulated the placement problem to satisfy constraints as an instance of the NP-hard VSBPP bin packing problem. We have defined the residual resources and communication overhead as the objective functions for the placement and we have subsequently formulated the problem in CP and proposed a modified version of the Best-Fit Decreasing greedy algorithm as a polynomial time solution. We have implemented a one instance algorithm as a legacy approach. We have evaluated our approach against other Greedy algorithms, the one_instance approach and the CP solution. The results have shown that BFD significantly outperforms the other models up to 50% while satisfying the corresponding traffic and resource capacity constraints.

References

[1] A. Ali, R. Cziva, S. Jouët, and D. P. Pezaros, *SDNFV-Based DDoS Detection and Remediation in Multi-tenant, Virtualised Infrastructures*. Cham: Springer International Publishing, 2017, pp. 171–196.

[2] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[3] Silver peak. [Online]. Available: https://www.silver-peak.com/

[4] Snort. [Online]. Available: https://www.snort.org/

[5] Suricata. [Online]. Available: https://suricata-ids.org/

[6] A. Gember-Jacobson, R. Viswanathan, C. Prakash, R. Grandl, J. Khalid, S. Das, and A. Akella, "Opennf: Enabling innovation in network function control," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 163–174, Aug. 2014.

[7] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2008–2025, Aug 2017.

[8] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.

[9] A. Ali, C. Anagnostopoulos, and D. P. Pezaros, "On the optimality of virtualized security function placement in multi-tenant data centers," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[10] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *2015 IEEE 4th International Conference on Cloud Networking (CloudNet)*, 2015.

[11] F. Wang, R. Ling, J. Zhu, and D. Li, "Bandwidth guaranteed virtual network function placement and scaling in datacenter networks," in *2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC)*, Dec 2015, pp. 1–8.

[12] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *2010 Proceedings IEEE INFOCOM*, March 2010, pp. 1–9.

[13] P. Bereziski, B. Jasiul, and M. Szpyrka, "An entropy-based network anomaly detection method," *Entropy*, vol. 17, no. 4, 2015.

[14] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 219–230.

[15] T. Huang, H. Sethu, and N. Kandasamy, "A new approach to dimensionality reduction for anomaly detection in data traffic," *IEEE Trans. on Netw. and Serv. Manag.*, vol. 13, no. 3, pp. 651–665, Sep. 2016.

[16] L. Huang, X. Nguyen, M. Garofalakis, M. I. Jordan, A. Joseph, and N. Taft, "In-network pca and anomaly detection," in *Advances in Neural Information Processing Systems*, 2007, pp. 617–624.

[17] R. Keralapura, G. Cormode, and J. Ramamirtham, "Communication-efficient distributed monitoring of thresholded counts," in *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '06. New York, NY, USA: ACM, 2006, pp. 289–300.

[18] J. B. Predd, S. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56–69, 2006.

[19] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, 2012, special Section: Energy efficiency in large-scale distributed systems.

[20] D. S. Johnson, *Bin Packing*. New York, NY: Springer New York, 2016.

[21] K. Fleszar and C. Charalambous, "Average-weight-controlled bin-oriented heuristics for the one-dimensional bin-packing problem," *European Journal of Operational Research*, vol. 210, no. 2, 2011.

[22] W. Leinberger, G. Karypis, and V. Kumar, "Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints," in *Parallel Processing, 1999. Proceedings. 1999 International Conference on*. IEEE, 1999, pp. 404–412.

[23] Cplex. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio/