A Formal Approach to Verify Connectivity and Optimize VNF Placement in Industrial Networks

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

A Formal Approach to Verify Connectivity and Optimize VNF Placement in Industrial Networks

Guido Marchetto, Riccardo Sisto, Fulvio Valenza, Jalolliddin Yusupov, Adlen Ksentini

***Abstract*—The increased flexibility and inter-connectivity of modern industrial communication networks, obtained through the use of innovative technologies like Network Function Virtualization (NFV) and Software Defined Networking (SDN), requires a secure and manageable framework to support the new communication and computing needs. To focus on these requirements, this paper proposes a framework for reliable placement of services across physically separated locations, which offers both system optimization, in terms of latency and resource utilization, and connectivity policy enforcement to guarantee service reliability, safety, and security. This is achieved by exploiting a new approach to solve the virtual network embedding problem, using Optimization Modulo Theories (MaxSMT), which allows the use of very expressive constraints.**

***Index Terms*—industrial network, reachability verification, virtual network embedding**

I. Introduction

Industry 4.0 is nowadays paving the way to modern intelligent Industrial Network Systems (INSs) sharing technical ambitions. This includes the ability to manage more heterogeneous communications among smart devices (e.g., smart manufacturing, smart factory, and smart energy) or reconfigure the system on-the-fly in an efficient and cost-effective manner. The massive adoption of interconnected computer-based networks (i.e., the Industrial Internet of Things) allows dealing with goods and services in an efficient and inexpensive way, while making the remote control, monitoring, and management of physical systems easier and cheaper. Network Function Virtualization (NFV) is certainly one of the main candidates to support this need for high flexibility, whereas Software Defined Networking (SDN) allows to separate network control functions from network forwarding functions. These technologies allow rapid service provisioning, a great level of dynamicity, and abstraction of the offered features. Hence, NFV and SDN can transform current services of industrial networks, consisting of vendor-specific hardware implementations, into software implementations embedded in commodity servers, i.e., into software-defined INSs.

Complexity of service graphs with multiple endpoints and real-time (re)configuration performed due to the auto-scaling, migration and life-cycle management of network services, however, introduce the challenge of preserving the correct behavior of a continuously changing network in case of attacks, failures or maintenance tasks. Unfortunately, network administrators manage network and security functions through the manual configuration of low-level parameters. This manual approach becomes almost impossible to apply in the context of continuously changing virtual networks, because of the difficulty of envisioning, in real-time, the correctness of the whole configuration of such large systems, guaranteeing an adequate level of protection. Without an alternative automated approach, there would be an increasing risk of exposure to cyber threats targeting industrial endpoint devices, such as Remote Terminal Units, Advanced Metering Infrastructure, Process Automation Controller, Human Machine Interfaces (HMIs), with possible negative effects on both human safety and business. In fact, security and safety are generally interwoven in INSs, as cybersecurity-related incidents potentially could result in multiple fatalities or environmental disasters. For example, security flaws in INSs that monitor or control "critical" industrial processes (e.g. water, gas, energy distribution, nuclear engineering farm) may be catastrophic.

This paper reflects our contributions and their importance to help to solve the above issues with a suitable automated tool based on sound theoretical foundations that can prevent misconfigurations, conflicts or sub-optimizations in the network.

On the basis of a preliminary work [1], in this paper we present a full formal approach that automatically verifies the connectivity properties of industrial networks and generates optimal placement plans for mapping virtual machines to physical machines. With respect to the preliminary paper, this paper provides a more general formulation of the problem and it introduces a new policy model, which supports multiple forms of reachability invariants. In particular, the proposed verification and Virtual Network Embedding (VNE) problems are formulated in First-Order Logic and solved jointly by exploiting the Optimization Modulo Theories (MaxSMT) approach. To the best of our knowledge, no other solutions exist in the INS context that solve both the problems of VNE and formal analysis of network properties in "one-shot", by combining them together. The rest of the paper is structured as follows. It starts with a presentation of related work in Section II, followed by the motivation and presentation of our approach in Section III. The service request model is described in Section IV, while the next two sections (Section V and Section VI), provide the details of how the formal verification and VNF placement problems are solved jointly. In Section VII the experimental evaluation of our approach and a use case are presented. Section VIII concludes the paper.

II. Related Work

Most of the past work formulates the embedding problem using combinatorial approaches such as mathematical pro-

G. Marchetto, R. Sisto, F. Valenza, J. Yusupov, are with the Politecnico di Torino, DAUIN; ({first.last}@polito.it), A. Ksentini are with the Eurecom (adlen.ksentini@eurecom).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

gramming (MP). However, MP is limited to arithmetic expressions over integer, binary, or real variables [2] and it is specific to a class of problems whose formulation satisfies certain mathematical properties (e.g., integer programming, mixed-integer linear programming). It is important to mention that there exist work on a transformation of propositional calculus statements into integer and mixed-integer programs [3], [2], which then can be solved by commercial solvers (e.g., CPLEX, Gurobi, LINDO, LINGO) efficiently. At the same time, these approaches acknowledge that the transformation is impractical in most cases and often unable to generate MaxSMT encodings for many of the instances, due to the huge amount of produced clauses. For this reason, existing approaches presented below, deal with the placement and verification problems separately and, to the best of our knowledge, no one proposes a unified modeling approach of the two problems. With this respect, the most related work to our one is arguably VFenceSynth [4], which encodes the placement and verification synthesis problems into SMT logics and provides a feasible solution with a model that is the placement plan. However, it is not the optimal placement plan, which may result in overutilization of the resources in servers. The next subsections provide an overview of previous related work on these separate subjects.

A. Formal verification

Recent work has made great progress for formal automated network configuration verification tools ([5], [6], [7], [8], [9]) and a few related to forwarding behavior of industrial networks is highlighted. We adopt and enrich the notion of reachability from Verigraph [5], which provides a method for modeling service graphs as sets of logical formulas and for verifying satisfiability of these formulas. One of the biggest enrichments is the MaxSMT formalism, which allows us not only to verify network invariants, but also solve the placement problem of network nodes in an optimal way. In the context of industrial control systems, [6] proposes verification of the integrity of the message flow in two industrial protocols, with a modeling approach that is very similar to the one of Verigraph [5]. Instead, the authors of [8] focus on formal validation of reliability requirements of protection functions, which are limited to protection relays, instrumentation (voltage, current, frequency, and other sensing devices) and Intelligent Electronic Devices. Similarly, [7] performs a formal analysis of security and resiliency in smart grid scenarios and validates the modeled system against resiliency specifications. This approach however only takes into consideration SCADA devices while it does not address complex service graphs, where the network consists of many network devices, as in future INSs.

B. VNF placement

In addition to the aforementioned literature, the related work on VNE is covered and our approach is compared to the state of the art. These approaches are categorized into exact ([10], [11], [12], [13], [14]) and heuristic-based ([15], [16], [17]) solutions. The purpose of VNE is to find the optimal solution with a particular objective. There is a further classification on the list of related work depending on whether

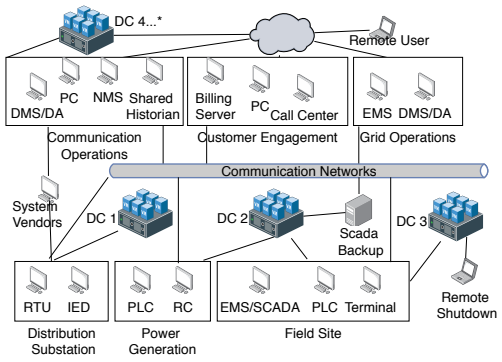


Fig. 1: An illustrative view of an INS

the minimization of the end-to-end delay is the main objective ([12], [18], [16], [17], [14]) or not ([10], [11], [15]). In fact, low latency communications are the premise for industrial applications, where mission- and time- critical systems require an end-to-end delay of the order of 1-100 ms ([19], [20]). An optimization algorithm is presented in [17] for the virtual network embedding problem over a set of distributed substrate nodes taking into account the maximum allowed end-to-end delay. According to [21] maximum allowed tolerance for the session establishment in SCADA systems is around 75 sec, which is not optimal for real-time applications where the connection must be established as fast as possible. Thus, the adaptation to lower values is recommended. The authors only solve the placement and routing problems for each path in a service request independently. Instead, our tool accepts in a single service request the whole service graph consisting of multiple paths.A Linear Programming (LP) optimization problem is formalized in [18] for minimizing the number of security functions. But this work fails to investigate resource-constrained scenarios of the placement problem. The authors of [22] propose a novel policy language to improve the expressiveness of the network properties by solving the placement problem with one limitation. As noted at the beginning, the mixed-integer formulation used in this approach is still not expressive enough and it does not model the actual forwarding behavior of the network. Instead, the authors rely on the assumption that the VMs where the functions are deployed are trustworthy and ensure correct packet forwarding among the containers and the external network.

III. Problem Statement

This paper proposes a methodology to jointly perform, in an automatic way, connectivity policy verification and optimal VNE in INSs. This section first introduces and motivates our work, by means of motivating use cases. Then, it provides a high-level description of the adopted approach, which will be further developed in the rest of the paper.

A. Motivating use cases

A possible infrastructure supporting the software-defined INS paradigm is shown in Fig. 1. Here a bare set of endpoint hardware devices of SCADA systems fixed in the power

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

plant is considered and interconnected by means of physical network topology. Physical network topology is provided by NFV infrastructure for the hosting of virtualized network functions and SDN allowing to separate the control and data for advanced use cases. In this paper, the focus is on distributed INS systems known as Smart Grids, even though our approach is generic for all industrial systems. In the case of non-distributed industrial systems, only consideration is on a single domain NFV infrastructure as a single DC with multiple nodes (i.e., from different providers such as Amazon AWS service, Microsoft Azure, and also OpenStack based), where the NFV orchestrator is in charge of the availability of the resources.

In our use case, a SCADA system of software and hardware elements is used to control and monitor industrial processes in a smart grid. The smart grid consists of distributed substations and field sites with intelligent devices and sensors, whereas SCADA control centers, defined as Communication Operations and Grid Operations in Fig. 1 are equipped with host computers and servers, where an engineer or operator can supervise the process, as well as receive alarms from the power grid. In this scenario, an incoming service request - Service Function Chaining (SFC) as defined by Internet Engineering Task Force (IETF), also known as VNF-Forwarding Graph (VNF-FG) in ETSI [23], might involve pairs or lists of these industrial endpoint devices, together with required network functions in between these nodes, thus forming a service chain or graph. A set of network functions forming a service graph is deployed in an infrastructure consisting of one or more data centers. The service request can also include the configuration of each network function and, last but not least, a number of policies that an administrator could specify to check the correctness of the network behavior resulting from the service request. This is the main criteria of the flexible but critical environment where security errors may damage the entire system. As an example of a security threat in the industrial field, firewall misconfiguration can introduce high risk, where firewall table may include only inbound traffic policies and grant any outbound traffic, neglecting the very real possibility that an attacker could be located inside the industrial zone attempting to communicate outwards, looking to acquire information about control system functions, configurations, and operations. Usually, these types of misconfigurations are detected by thoroughly analyzing the policies and firewall rules of multiple network devices after the services are deployed. Instead, this paper proposes a novel technique, which allows to detect those misconfigurations before the actual deployment of a service and provide a formal assurance that the rules in the network will satisfy all policy requirements of the user.

Two examples are illustrated in Fig. 2, which represents two possible service graphs that can be deployed in the above described infrastructure (VNFs are represented as cubes). In the first scenario, depicted in Fig. 2(a), the service graph implements an industry recommended network security guideline, where the Power Generation field industrial control systems must be isolated from a Distribution Substation. This is done by means of a firewall. The graph also includes an AMI linked to a back-end server, for instance, a Billing Server and, in order to improve the overall security of the system, a DPI module.

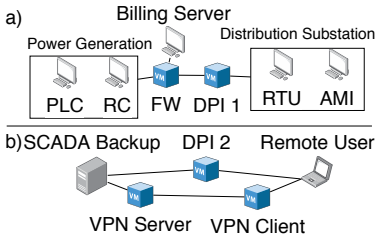


Fig. 2: a) rechability and isolation b) alternative path

In this case the service request may include two connectivity policies to be verified: clearly, the isolation between the Power Generation field and the Distribution Substations, but also the reachability between the Billing Server and the AMI device. Those can be satisfied or not depending on the specific configuration deployed in every single VNF and device. These two connectivity policies defined by the user can be satisfied, only if configuration parameters of intermediate VNFs allow this. In the second scenario, shown in Fig. 2(b), the service graph represents a reliable SCADA backup system where the administrator requires the SCADA backup endpoint be reachable by means of two redundant paths with different security features. Those are provided by a DPI function in the former and by a VPN tunnel in the latter. The administrator can, therefore, be interested in verifying that the service request as given satisfies an alternative path property, i.e. that, with the deployed graph and configurations, the target is actually reachable by means of the two paths.

In both cases, and in general in the industrial field, latency minimization or bounding is key to guarantee proper operation and performance, while taking care of datacenter utilization is important for several reasons ranging from cost reduction to power saving. This is the reason why we focused on these two parameters for the placement task of our framework, which however can be easily extended to optimize placement according to different elements.

B. Proposed Framework

In order to reach our verification and optimization targets, we developed a comprehensive framework depicted in Fig. 3 which can solve the joint VNE and formal verification problem that was outlined, by means of a MaxSMT solver. The input is a service request, which includes all the information provided by the user (the network graph with VNF instances and their configuration parameters, and the network policies). The details of this request are given in Section IV.

Our approach for solving the problem is to formulate it as a MaxSMT problem, which receives two sets of FOL clauses as input: hard and soft clauses. Hard clauses must be true necessarily, while soft clauses may be true or false, and they are associated with weights. The problem consists of finding an assignment of free variables that makes all hard clauses true and that maximizes the total weight of the soft clauses that are true. From the service request, an automated clause generator generates the hard and soft clauses that are then given as input to the MaxSMT solver (we use z3, a well-known solver). In

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

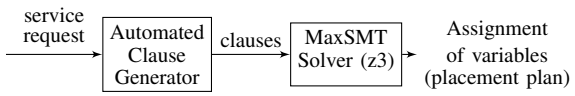


Fig. 3: Functional diagram of the proposed approach.

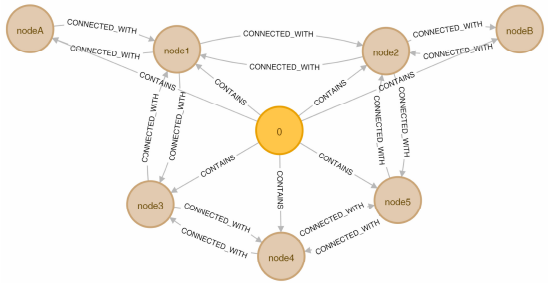


Fig. 4: A placement plan generated by the framework. Red nodes are VNFs and brown nodes are the substrate nodes.

our approach, hard clauses are used to model the forwarding behavior of VNFs, the connectivity-related network policies to be verified, and the hard constraints associated with an allocation (e.g., the overall latency of a path between the start and end points of a flow should not exceed the maximum tolerable latency). These clauses ensure that, if a solution is found, it satisfies all user requirements.

In modeling the placement problem, soft clauses play an important role. For instance, the decision to use a particular physical link to connect two VNFs is defined as a soft clause, which can be falsified in favor of choosing another possible link, i.e. in favor of a similar other soft clause. Weights may correspond to a number of metrics in the optimization problem. As said, this paper uses the average latency of the substrate links and the cost of using specific data centers in an NFV infrastructure as weights, because of their interest in industrial systems, but other similar criteria can be used. As a result, a multi-objective problem is addressed: minimizing the end to end latency and minimizing the total utilization cost of the NFV infrastructure. In order to overcome this competition between objectives, we prioritize them with the help of different weights, as discussed in Section VI.

If the solver finds a solution, it returns the found assignment of variables, which is a possible plan for our placement problem. Our framework converts this assignment to a graphical representation as shown in Fig. 4, which can be accessed using the GUI of our tool. If there is no way to satisfy all the hard clauses, the solver provides a non-satisfiability report, which means the input has to be fixed.

IV. Service Request Model

The verification and placement problem requires, as input from the user, the Service Graph (SG) and the connectivity policies to verify. This section elaborates on each of them.

A. Service Graph.

The SG is modeled as a directed graph $G^v = (V^v, L^v, A_V^v, A_L^v)$, similar to previous work in [24], [25]. V^v is the set of vertices, partitioned into two disjoint subsets: N^v and E^v. N^v is the set of VNFs n^v to be allocated, while E^v is the set of endpoint VNFs e^v, whose allocation on the substrate network is assumed to be fixed. Notice that the endpoint devices do not necessarily have to be virtualized and they can co-exist as hardware solutions without affecting our models. L^v is the set of edges (representing the links that connect VNFs with one another). Each edge has a direction which means that outgoing and incoming packet flows are associated with different edges. Finally, A_V^v and A_L^v are the sets of properties that can be taken by the attributes of vertices and edges, respectively. Although this model is general enough, allowing various possible attributes such as CPU, memory and link latency, for simplicity this paper considers only a limited set of properties. Each VNF n^v to be allocated has a required storage capacity attribute (i.e., the storage required by a VNF), denoted storage(n^v), a fixed packet processing delay attribute (i.e., time required to process the incoming traffic), denoted lat(n^v), and a functional type attribute (i.e., software program), denoted func(n^v), where $n^v \in A_V^v$. Depending on the functional type of the VNF, a user may also define the configuration parameters conf(n^v) of that specific VNF, which is another attribute used to define functional characteristics of each VNF in the service graph. All these VNF attributes, here represented as functions, are provided with the SG. Processing delay is fixed as we assume "no congestion", due to the fact that endpoint devices have low load values in industrial networks. Processing delay is inversely proportional to the computational power of the VNF and it is a property of the VNF and not a demand. For what concerns links between nodes, a bandwidth property denoted as band(l^v) is considered. Finally, vertices are uniquely identified by means of integer indexes, where v_j^v denotes the vertex with index j. Depending on the type of vertex v_j^v, it may correspond to a service node n_j^v or service endpoint e_j^v.

B. Policy model

In our model, a connectivity policy rule p is a tuple $p = (t, s, d)$, where s and d represent the source and the destination of a communication, respectively, and t is the type of policy described below. Source and destination s and d are both subsets of SG vertices ($s, d \subseteq V^v$). They can include either single VNFs or zones (e.g., whole IP subnets). For what concerns the type t, in this paper we consider it can take one of three different possible values, but our approach is flexible enough to accommodate other types. In particular, we consider reachability (R), isolation (I), and the presence of alternative (A) paths. There is reachability from network node/zone s to d, if there exists at least one path that connects s to d and in this path, there are no functions that block this communication. Instead, there is isolation from s to d, if there is no path that connects s to d, or there are paths but in each one of them there is at least one function that blocks this communication. Finally, there are alternative paths from s to d, if there are at least two disjoint paths that connect s to d and that do not include functions that block this communication. By disjoint paths, we mean paths that have no shared edge and vertices.

With reference to the previously described Fig. 2, the properties introduced in Section III can be expressed by means of this notation. In particular, reachability between the Billing Server and the AMI device defined in scenario (a) can be expressed as: p_{a_1} = (R, {Billing Server}, {AMI})
whereas the isolation rule that isolates two zones can be expressed as: p_{a_2} = (I, {PLC, RC}, {RTU, AMI})

Furthermore, the policy on path alternative required in scenario (b) can be expressed in the following way:

$$p_b = (\text{A}, \{\text{Remote User}\}, \{\text{SCADA backup}\}) \quad (1)$$

V. Formal Verification

This section presents our formal methodology for the verification of connectivity policies. Given the input SG and policy rules, as presented in the previous section, our framework constructs a set of FOL formulas that model the forwarding behavior of the SG and then, using these formulas, it verifies each policy rule as explained below.

A. Network model

The forwarding behavior of the network is modeled by tracking the packets that each node can send and receive. A packet is abstracted as a list of fields. Here are some field examples for an IP packet: (i) src and dst are the source and destination addresses; (ii) proto is the protocol type; (iii) origin is the endpoint that originally generated the packet. The specific fields to use can be easily customized as needed. We introduce an abstract field of a packet to store the actual origin, which is required in presence of network functions that modify packet headers. To retrieve information about the packet fields or the network nodes, we use function symbols. In FOL, function symbols do not state facts and do not form sentences. They are assigned any interpretation that is compatible with the constraints over the function. Examples of function symbols adopted in our model are Int sport(p) and Int dport(p), representing the source and destination ports of packet p used for TCP/IP networking at the application layer, respectively. The forwarding behavior of the network is modeled in terms of two predicates: • send(n_0, n_1, p), which is true if source node n_0 can send packet p to destination node n_1, with a constraint defined below, stating that source and destination addresses in the packet must be different; • recv(n_0, n_1, p), which is true if destination node n_1 can receive packet p from source node n_0, with the same constraint on source and destination addresses. The following set of conditions imposed on the above functions expresses the general abstraction of the network forwarding behavior:

$$\begin{aligned}\text{send}(n_0, n_1, p_0) \implies (n_0 \neq n_1 \wedge p_0.\text{src} \neq p_0.\text{dst} \wedge \text{sport}(p_0) \geq 0 \\ \wedge \text{sport}(p_0) < \text{MAX\_PORT} \wedge \text{dport}(p_0) \geq 0 \\ \wedge \text{dport}(p_0) < \text{MAX\_PORT}), \forall n_0, n_1, p_0\end{aligned} \quad (2)$$

$$\text{recv}(n_0, n_1, p_0) \implies \text{send}(n_0, n_1, p_0), \ \forall n_0, n_1, p_0, \quad (3)$$

where (2) states that source and destination nodes (n_0 and n_1), as well as source and destination addresses in the packet (p_0.src and p_0.dst) must be different (the dot notation is used to access packet fields), and also the constraints that source and destination ports must be within a valid range of values. Finally, (3) states that if packet p_0 can be received by node n_1 from node n_0, then p_0 can also be sent by n_0 to node n_1.

B. VNF models

Industrial networks are composed of switches and more complex network functions, such as firewalls, which may alter the overall forwarding behavior of the network in a way that depends on their configuration rules. Hence, we introduce another set of formulas that describe the abstract forwarding models of such network devices. For what concerns the endpoint industrial devices (e.g., AMI, PAC, VFD, HMI, and RTUs), they can send/receive packets to/from some other endpoints. In this work, for simplicity, the configuration parameters of these devices are not taken into account. The other network nodes are the so-called middleboxes, i.e. the VNFs that are on the paths between endpoints. Each one of them needs specific formulas, unless it acts simply as a packet forwarder, in which case its model can be omitted. Due to space limitations, only the model of a stateless firewall is presented, but we developed models for several other types of VNFs, in a way similar to what can be found in [5]. The cost of developing new VNF models is not negligible. However, most of the commonly used VNFs belong to well known types, for which a catalog of models can be provided. Adapting a model found in the catalog to the simple variations of a specific VNF is more affordable. Moreover, there are ad hoc tools [26] designed for automatically extracting verification models starting from an abstract programming-language-like representation of a given network function, which lowers the cost of developing new VNF models.

Firewalls are security functions that impose a boundary between multiple domains in the network. They are designed to allow or block network connections based on specific rules. The forwarding behavior of a stateless firewall is modeled on the basis of predefined decision rules, that are configured when the service model is initialized. In particular, the decision rules are managed through the uninterpreted function fw_rule(l, m, n, o, p), which filters on the IP 5-tuple l, m, n, o, p. For example, if the firewall table is configured with the only rule <l_1, m_1,*,*,*>, the interpretation of the function is given by (4).

$$\text{fw\_rule}(l, m, n, o, p) == (l == l_1 \wedge m == m_1), \forall (l, m, n, o, p) \quad (4)$$

By default, a "whitelist" policy is used for our firewall model, but the "blacklist" policy can be easily set with a negation of the function. The model includes two clauses: if the firewall can send a packet, then it must also be able to receive the same packet from its predecessor, and fw_rule must return true for this packet:

$$\begin{aligned}\text{send}(n_{fw}, n_0, p_0) \implies (\exists (n_1) | \text{recv}(n_1, n_{fw}, p_0) \wedge \\ \text{fw\_rule}(p_0.\text{src}, p_0.\text{dst}, \text{sport}(p_0), \text{dport}(p_0), p_0.\text{proto}), \forall (n_0, p_0)\end{aligned}$$

Other functions can be modeled in a similar way, by variations of this formula. For example, in the case of a DPI, which performs application layer packet filtering, it is possible to model a lookup in a table of blacklist items by means of a function and to use a formula similar to the one of the firwall, but with the fw_rule function substituted by this one.

C. Policy enforcement

All types of policies supported by our tool (i.e., reacheability, isolation, and alternative path, as presented in Section IV) can be verified by verifying the following basic formula one or more times:

$$\exists(n_0, p_0) \mid \text{recv}(n_0, d, p_0) \wedge p_0.\text{origin} == s, \tag{5}$$

which means that node d can receive a packet that was originally sent by node s. Note that, according to this formula, node d may receive a packet that is different from the one originally sent, which takes into account the possibility that intermediate VNFs modify packets during their trip to destination. This is why only the origin of the received packet (p_0) is constrained ($p_0.\text{origin} == s$) while the other fields are not. Formula (5) expresses a reachability policy between two VNFs. An isolation policy is simply its negation, while an alternative path policy $p(A, s, d)$ can be verified as follows: first, all the pairs of disjoint paths connecting s to d in the SG are searched by means of graph algorithms. For each one of these pairs, reachability is checked separately for each of the two alternative paths that make the pair. This is done by restricting the graph to the path being considered and by applying (5). As soon as a pair is found for which both paths satisfy reachability, we know the policy rule is true, otherwise it is not satisfied.

VI. VNF Placement

This section formalizes the optimization problem related to the placement of VNFs in the substrate network. We first define a model for the substrate network, then the FOL formulas that model resource allocation constraints, routing tables and optimization objectives.

A. Substrate graph.

We model the substrate network, which is fixed for our tool, as another undirected graph similar to the one that describes the service request (see Section IV): $G^s = (V^s, L^s, A^s_V, A^s_L)$, where $V^s = N^s \cup E^s$ is the set of vertices, made up of the two disjoint subsets N^s (substrate nodes) and E^s (substrate endpoints), L^s is the set of edges (i.e. the links connecting substrate nodes and endpoints with one another), instead A^s_V and A^s_L are the sets of values that can be taken by the attributes of the vertices and edges, respectively. In this paper, v^s ranges over V^s, n^s ranges over N^s, e^s ranges over E^s, and l^s ranges over L^s. As vertex attributes, we can use, for example, CPU, RAM, hard disk capacity, geographical location, etc., while for link attributes, throughput, latency, jitter, etc. can be used. For simplicity, here only one substrate node attribute (hard disk capacity) and only one link attribute (latency) are retained, but the extension to other metrics is straightforward. As industrial networks are geographically distributed, propagation delay can affect time-critical communications. For each substrate node n^s, its associated available storage capacity is denoted $\text{storage}(n^s) \in A^s_V$, while for each substrate link l^s, its associated latency is denoted $\text{latency}(l^s) \in A^s_L$. Similarly to what has been done with the service request graph, integer indexes are used for the vertices of this graph too, with the same notation. Moreover, l^s_{jk} denotes the link between the vertices indexed by j and k.

B. Resource allocation.

This subsection describes the hard constraints related to resource requirements of the VNE problem. First, we define boolean variables y_i and x_{ij} with the following meaning: y_i means that substrate node n^s_i is in use while x_{ij} means VNF n^v_i is allocated on substrate node n^s_j. The notation $n^v_i \uparrow n^s_j$ for x_{ij} is also used. Then, we use two mapping functions in order to represent the mapping of a service request onto the substrate network: M_n maps the VNFs of the service request onto the substrate nodes, in such a way that their resource requirements are satisfied, while M_e maps endpoints. The formal definition of M_n is:

$$M_n(n^v) = n^s, \quad \forall n^v \in N^v, n^s \in N^s, n^v \uparrow n^s \tag{6}$$

subject to

$$\sum_{i | n^v_i \uparrow n^s_j} \text{attribute}(n^v_i) \cdot x_{ij} \leq \text{attribute}(n^s_j) \cdot y_j, \quad \forall j | n^s_j \in N^s \tag{7}$$

with the assumption that, for x_{ij} and y_j, values true and false correspond to 1 and 0, respectively. Equation (7) defines the generic constraint on the generic attribute function $\text{attribute}(n)$, which stands for any one of the specific attribute functions, such as $\text{storage}(n)$ and $\text{cpu}(n)$. Hence, (7) represents a set of constraints meaning that the sum of the required values of each attribute (e.g. storage and CPU) for the VNFs mapped onto a substrate node cannot exceed the value of the same attribute available on that node. Behind this rule there is the underlying assumption that VNFs from the same service request can share the same substrate nodes, which is commonly allowed in NFV systems, e.g. as a way to reduce latency. The representation of M_n is built as a set of clauses including the inequalities in (7) plus additional clauses that are necessary in order to state that M_n must be a function. More precisely, these clauses state that M_n maps each VNF exactly onto one substrate node:

$$\sum_{j | n^s_j \in N^s} x_{ij} = 1, \quad \forall i | n^v_i \in N^v. \tag{8}$$

Finally, the following clauses (implications) are necessary in order to correctly relate y_j to x_{ij},:

$$y_j \implies \bigvee_i x_{ij}, \quad \forall j | n^s_j \in N^s, \tag{9}$$

The meaning of these last clauses is that if substrate node n_j is used, then there must be at least one VNF mapped onto it.

C. Routing tables.

The concept of transfer functions, which was previously developed by VeriFlow [27] and HSA [28], is used to model the network forwarding behavior of the virtual service. A transfer function represents the routing tables of a network function involved in the service request. By means of the formulas corresponding to the transfer function, we describe the path packets take towards their final destination. This allows us to construct service graphs and integrate the presented models of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

network functions to the overall network model. According to the input SG, for each VNF v_i^v and its adjacent one-hop neighbor v_a^v, our tool automatically generates a predicate route(v_i^v, v_a^v, l_{jk}^s). The predicate is true if the adjacent neighbor of v_i^v is v_a^v and it can be reached via the link that connects substrate nodes j and k, i.e. l_{jk}^s.

D. Optimization Objectives.

By means of the forwarding behavior of VNFs, the placement constraints can be formulated and a weight is assigned to different placement plans. A decision to place the adjacent node to an endpoint VNF e_n^v in the SG is represented by a set of soft clauses with a negative weight, whose absolute value is the link latency. This will cause the MaxSMT solver to minimize the total latency of the chosen infrastructure path. For each possible substrate node n_k^s onto which the adjacent neighbor VNF in the SG (n_a^v) can be allocated, the following soft clause is added:

$$\text{Soft}((\text{route}(e_0^v, n_a^v, l_{0k}^s) \implies x_{ak}), -\text{lat}(l_{0k}^s)), \tag{10}$$

where the notation Soft(c, w) means that c is a soft clause with associated weight w. Note that the location of e_0^v is fixed in the substrate endpoint e_0^s

In practice, the routing table of the endpoint VNF specifies to which substrate node k a packet is forwarded depending on the allocation of the next VNF in the SG.

The soft clauses for the other VNFs $n_i^v \in N^v$ in the SG, with $i > 0$, are generated similarly:

$$\text{Soft}((\text{route}(n_i^v, n_a^v, l_{jk}^s) \implies x_{ij} \wedge x_{ak})), -\text{lat}(l_{jk}^s)), \tag{11}$$

i.e., if VNF i forwards packets to the adjacent VNF a in the SG through link l_{jk}, then the boolean variables x_{ij} and x_{ak}, which indicate the locations of the VNFs, must be true. If $j = k$, i.e. two VNFs are mapped onto the same substrate node, then $\text{lat}(l_{jk}^s) = 0$, and a soft clause with weight equal to zero is added. The introduction of the variables used to indicate locations of the VNFs allows us to present the bandwidth constraint that has to be considered to avoid overloading of substrate link capacity.

$$\sum_{i|n_i^v \uparrow n_j^s} \text{band}(l_{ia}^v) \cdot (x_{ij} \wedge x_{ak}) \leq \text{band}(l_{jk}^s), \quad \forall j, k | n_j^s, n_k^s \in N^s \tag{12}$$

Since for each VNF it is also possible to specify a (fixed) processing delay, represented by the lat(n^v) function, this must be considered when computing the total end-to-end latency. In particular, if one is interested in specifying an upper bound on the total end-to-end latency that must be guaranteed in the system, one can formulate this constraint as an additional hard clause. Furthermore, we need to properly manage our multi-objective optimization problem, as described in Section III, i.e., to give priority to latency minimization rather than to efficient resource utilization. In order to do that, the Lexicographic Multi-Objective Programming (LMOP) problem [29] can be encoded into MaxSMT using the Boolean Lexicographic Optimization scheme described in [30], by assigning weights to each objective function, where the objectives can be ranked in order of importance. As already noted, the solver will minimize latency because of the weights associated with the soft clauses for the route predicates. As we want to minimize also the number of used substrate nodes, an additional soft clause is generated for each substrate node $n_i^s \in N^s$: Soft$(\neg y_i, L)$, where L is a constant selected according to the target of the minimization: a larger L gives priority to node utilization minimization, whereas a smaller L gives priority to latency minimization. The MaxSMT solver attempts to assign false values to the boolean variables y_i in order to minimize the penalty for falsified clauses in the current model, thus minimizing the number of nodes in use. Then, by feeding the MaxSMT solver with the conjunction of the clauses expressing the forwarding behavior of the network (Section V) and the ones representing the placement constraints (Section VI), we obtain, at the same time, the verification that the specified policies hold, and the optimal placement plan, or an indication that the policies are not satisfied.

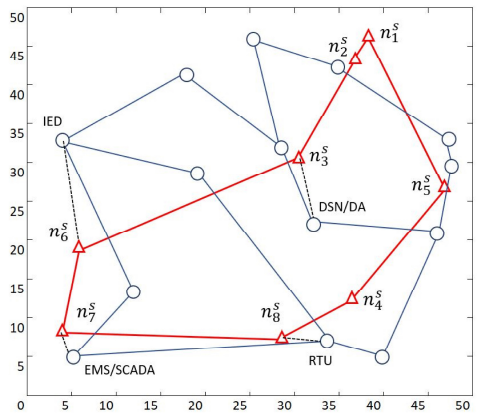


Fig. 5: Experimental topologies, where the location of endpoint smart grid nodes is fixed. Physical infrastructure composed of △ 7 DCs and ◯ IEEE 14-bus power system

VII. Implementation and Validation

To evaluate our approach, a smart grid is considered as a representative use case of an industrial network. In particular, we use the South African National Research Network (SANReN) [31] network topology as the substrate backbone with 7 nodes and 7 links for the smart grid nodes as shown in Fig. 5. The IEEE BUS 14 test system topology is chosen as a power grid infrastructure, while a simplified control network consisting of smart grid endpoints is built based on a real topology (not referenced because of a nondisclosure agreement). These topologies are placed in a 50x50 square kilometer geographic area as shown in Fig. 5 and the Euclidean distance between nodes in the coordinate system is considered as a metric of the link latency. All experiments run on a single CPU core of an Intel i7-6700 based PC, with 32GB RAM.

There are 19 buses/nodes in the substation locations, 17 connections between buses, 8 transformers and 11 constant impedance loads. Each power system bus functions as a gateway router that is connected to the closest substrate network node either through wired or wireless links (eg. CDMA, 4G/LTE). The gateway routers aggregate traffic from endpoint VNFs to be forwarded to other endpoints or substrate nodes

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

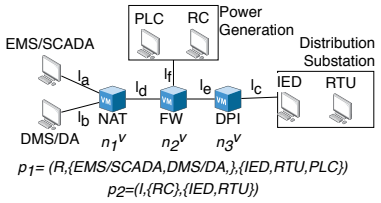


Fig. 6: Service request example

TABLE I: Computation time of different topologies

Topology	Nodes	Links	Time (O&V)	Time (O+V)
Internet2[33]	10	13	0.3s	0.54s
GEANT[33]	22	36	12.9s	13.2s
UNIV1[34]	23	43	21s	23.23s
AS-3679[35]	79	147	24.3s	26.2s

throughout the substrate network. For the sake of simplicity, only the part of the power network topology and preset/map the endpoint VNFs of the service on them are highlighted in Fig. 5. The physical connections that link the endpoint VNFs to substrate nodes are represented with dashed lines.

As an example of a service request, the paper focuses on the use case represented in Fig. 6, which includes SCADA control components, peripheral devices required to interface with the power grid and machinery, and network functions to be allocated in between. In this example, there is an assumption that EMS/SCADA, DMS/DA (distribution management system and distributed automation), and devices from Power Generation and Distribution Substation are the endpoint VNFs of the smart grid and the placement of these devices is predetermined and attached to specific data centers (i.e., they are not taken into account in solving the optimization problem). Control center endpoints are located behind the NAT n_1^v and connected to the SCADA peripheral devices through the Firewall n_2^v. Additionally, there is a DPI network function n_3^v that inspects the packets arriving at IED and RTU. Tests are run to obtain an optimal placement in terms of end-to-end latency and number of utilized data centers, while guaranteeing two policies are satisfied. In particular, to apply the reachability and isolation policies shown in Fig. 6, which state that the SCADA control devices (i.e., EMS/SCADA, DMS/DA) must be able to reach SCADA slaves in Power Generation station and in Distribution Station, whereas these substations must be isolated from each other to protect the network from unauthorized access. The service request also includes the following configurations for the involved VNFs, introduced by means of the notation described in Section IV: (i) NAT n_1^v: IP address range for devices in the private network; (ii) Firewall n_2^v: Allow traffic from NAT, deny others; (iii) DPI n_3^v: drop all packets containing some well-known attack signatures. The storage requirements of each VNF have been selected randomly with a uniform distribution between 1 and 10 gigabytes, while the available hard disk capacity of each data center has been selected in the range 10-15 gigabytes and does not contain any VNF, intially. Finally, the overall processing delay of each VNF is uniformly distributed between 5 and 10 milliseconds. [32].

The clauses are provided as an input to the solver, using the Java API. As the configuration parameters of the involved VNFs satisfying the two policies in Fig. 6, the z3 solver returns a model with values assigned to all variables x and y described in Section VI. In our use case, $x_{16}, x_{26}, x_{32}, y_6, y_2$. have value 1 (true). Under these circumstances, the minimum number of utilized DCs is equal to 2. Algorithms used by the solver in this work guarantee the optimality of the solutions that they produce. This is a latency-aware optimal placement plan of VNFs in substrate network for our use case, where the NAT n_1^v is allocated on the substrate node n_6^s, the FW n_2^v on n_6^s, the DPI n_3^v on the substrate n_2^s. Similarly, truth assignment of y_6 and y_2 shows that the substrate nodes n_6^s and n_2^s are in use.

Further, Table I presents (O&V column) the computation time obtained by feeding our tool with this service request as input and some well-known benchmark topologies as a substrate network. The first listed, Internet2 [33], is a backbone network consisting of 10 nodes with 100 Gb/s interfaces, supporting over 66,000 institutions in the United States. GEANT [33] is the high bandwidth European research and education backbone consisting of 22 nodes and 36 links. In order to check the scalability of our approach, UNIV1 [34] 2-tier campus data center network and AS-3679 [35] Rocketfuel ISP topologies with 23 and 79 nodes, are also used respectively. The case studies are based on the IEEE 14-bus power system for each topology of data centers. The time taken by the z3 MaxSMT solver to compute the results for the Internet2 topology adopted by [12] is only a few milliseconds. Computation time significantly increases for larger substrate networks, quickly exceeding 10 sec for substrate networks with ≥22 nodes. It must be noted also that the VNFs of the service request, in these scenarios, are configured to satisfy the connectivity policies. In case of a violation of a policy, our framework returns proof of the unsatisfiability of the problem without any placement plan, thus preventing problems from hitting the forwarding plane. For the sake of completeness, we also consider the performance of solving the optimization and the verification problems separately, i.e., the two models are processed separately on z3. As the variables related only to the optimization model of the MaxSMT instance have a larger number of degrees of freedom when there is no notion of policy verification (and vice versa), z3 solver spends more time to compute the instances of these two separate problems. This is confirmed by our experimental results presented in Table I (V+O column). It can be seen how the time required to compute the verification and optimization models separately is higher than solving them jointly. This further validates our approach of performing joint optimization and verification.

The scalability of our approach has been evaluated by measuring the time taken to solve problems of increasing size. The time spent by z3 to solve the MaxSMT instance for a single request with a varying number of DCs, VNFs, and policies is shown in Fig. 7. As it makes no sense to have large policy numbers with small VNF numbers and vice versa, the ratio of number of VNFs to number of policies is kept fixed at 1:2. Instead, multiple requests can be solved sequentially or even in parallel, by merging them into a single larger request. Parallelism can also be exploited for the solution of the MaxSMT problem as far as the solver supports it.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3002816, IEEE Transactions on Industrial Informatics

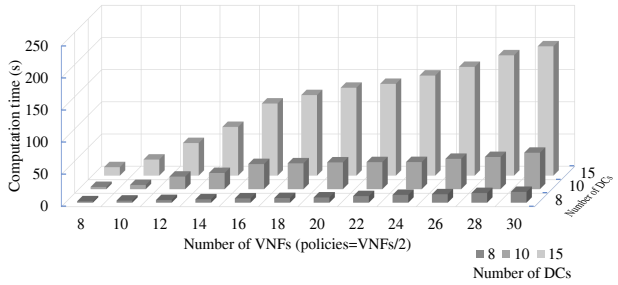


Fig. 7: Average execution time of the solver for each request

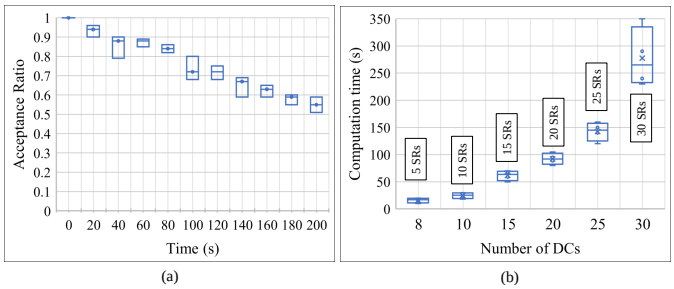


Fig. 8: (a) Acceptance ratio over time (b) Computation time.

In our case, the parallel disjoint tactics provided by z3 can be used. However, this analysis is out of the scope of this paper and remains to be investigated in a future study. We observe that computation time increases almost linearly with the size of the service request on the same substrate network. There is, however, a notable deviation in terms of computation time, when the number of DCs increases along with the number of VNFs and policies. This is due to the fact that for larger substrate networks, a larger set of allocation options has to be constructed, thus increasing the number of formulas given to the solver. As shown in Fig. 7, the average runtime remained within a minute for a moderately sized substrate network of 8 DCs, while for a large-scale network of 15 DCs, the z3 solver requires up to 3 minutes to generate a placement plan. In contrast, the only approach close to ours, VFenceSync [4], requires 10 minutes to obtain a non-optimal placement plan for the network consisting of 15 DCs. As the connectivity verification and network planning is usually done once and before the actual deployment, we argue that longer computational times can be permissible especially if they result in finding verified, optimal deployments that offer significant cost savings. Average Memory usage rises from 0.4GB to 0.6GB for the service request chain containing 5 VNFs and 10 respectively and continues to increase gradually to the peak memory usage of 1GB for the service request chain containing 14 VNFs. This shows how memory intensive the verification and optimization processes are.

Finally, the behavior of the proposed system when subject to multiple requests has been studied in simulated scenarios. Starting from an initial state with no service deployed and all resources available at time 0, the system receives a series of randomly generated requests. Of course, as far as new requests are successfully embedded, the probability of failure due to unavailability of the necessary resources increases. Fig. 8(a) shows the average acceptance ratio, i.e. the fraction of requests that are successfully embedded, over time. The average value is considered over 20 experiments due to the randomness introduced in defining the service requests. Another similar experiment has been made to study how computation time varies when the system receives an increasingly longer sequence of requests. Fig. 8(b) shows the total computation time required to embed a series of service requests (SRs), with a rate of 5 service requests arriving every 10 minutes. In order to keep the acceptance ratio of the algorithm equal to 1, the size of the substrate network is increased over the x-axis as the number of total requests increases. It is important to note that the network reconfiguration of a Smart Grid is performed not frequently, due to reasons such as weather factors, protection malfunction, service upgrade or cyber-attacks[36]. The service instantiation time of major NFV Orchestrators in case of a reconfiguration is usually in the order of minutes[37]. In comparison, the computation time introduced by our framework is acceptable.

VIII. Conclusions and Future Work

In this paper, a framework for efficient resource mapping and satisfaction of connectivity policies for industrial control systems that employ the NFV/SDN technology is presented. The combination of verification and placement problems enables latency-aware network embedding in the NFV infrastructure and allows us to check the end-to-end connectivity requirements of the network service in the presence of VNF configurations. To evaluate our approach, we have used the IEEE 14-bus system and several physical topologies of data centers. Our experiments show that the new MaxSMT approach can be used to find optimal solutions in order of seconds when considering substrate networks with significant numbers of data centers. Our work also shows the power of FOL formulations which allows the specification of expressive constraints compared to Integer Programming.

In future work, we will perform a Pareto analysis of the proposed optimization problem to show the trade-offs between different objectives. Other perspectives for future work include the online handling of service graphs, reliability and availability of the resources, in addition to allowing the migration of previously embedded service graphs.

References

[1] G. Marchetto, R. Sisto, J. Yusupov, and A. Ksentini, "Formally verified latency-aware VNF placement in industrial internet of things," in *14th IEEE Inter. Workshop on Factory Communication Systems, WFCS 18*. IEEE, 2018, pp. 1–9.

[2] E. Demirović, N. Musliu, and F. Winter, "Modeling and solving staff scheduling with partial weighted maxsat," *Annals of Operations Research*, vol. 275, no. 1, 2019.

[3] G. J. Gordon, S. A. Hong, and M. Dudík, "First-order mixed integer linear programming," *CoRR*, vol. abs/1205.2644, 2012.

[4] A. H. M. Jakaria, M. A. Rahman, and C. Fung, "A requirement-oriented design of nfv topology by formal synthesis," *IEEE Trans. Netw. Service Manag*, 2019.

[5] S. Spinoso, M. Virgilio, W. John, A. Manzalini, G. Marchetto, and R. Sisto, "Formal Verification of Virtual Network Function Graphs in an SP-DevOps Context," in *Proc. of the 4th European Conf. of Service Oriented and Cloud Computing (ESOC)*, 2015.

[6] J. Dreier, M. Puys, M.-L. Potet, P. Lafourcade, and J.-L. Roch, "Formally and practically verifying flow properties in industrial systems," *Computers & Security*, 2018.

[7] M. A. Rahman, A. H. M. Jakaria, and E. Al-Shaer, "Formal analysis for dependable supervisory control and data acquisition in smart grids," in *c Dependable Systems and Networks*, 2016.

[8] M. Masselot, S. Patil, G. Zhabelova, and V. Vyatkin, "Towards a formal model of protection functions for power distribution networks," in *Proc. of the Conf. of the IEEE Industrial Electronics Society*, 2016.

[9] C. Basile, D. Canavese, C. Pitscheider, A. Lioy, and F. Valenza, "Assessing network authorization policies via reachability analysis," *Comput. Electr. Eng.*, vol. 64, no. C, 2017.

[10] C. Basile, C. Pitscheider, F. Risso, F. Valenza, and M. Vallini, "Towards the dynamic provision of virtualized security services," in *Cyber Security and Privacy*, F. Cleary and M. Felici, Eds., 2015.

[11] C. Qian and S. S. Lam, "Greedy routing by network distance embedding," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, 2016.

[12] M. Hasan and H. Mouftah, "Cloud-centric collaborative security service placement for advanced metering infrastructures," *IEEE Trans. on Smart Grid*, vol. PP, no. 99, 2017.

[13] J. Aghaei, A. Baharvandi, A. Rabiee, and M. Akbari, "Probabilistic pmu placement in electric power networks: An milp-based multiobjective model," *IEEE Trans Ind. Informat.*, vol. 11, no. 2, 2015.

[14] J. Son and R. Buyya, "Latency-aware virtualized network function provisioning for distributed edge clouds," *Journal of Systems and Software*, vol. 152, 2019.

[15] 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 Trans. Netw.*, vol. 27, no. 1, 2019.

[16] M. Hasan and H. Mouftah, "Latency-aware segmentation and trust system placement in smart grid scada networks," in *Proc. of the IEEE 21st Int. Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD)*, 2016.

[17] M. Gharbaoui, C. Contoli, G. Davoli, G. Cuffaro, B. Martini, F. Paganelli, W. Cerroni, P. Cappanera, and P. Castoldi, "Experimenting latency-aware and reliable service chaining in next generation internet testbed facility," in *Proc. of the Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2018.

[18] M. Hasan and H. Mouftah, "Optimization of trust node assignment for securing routes in smart grid scada networks," *IEEE Syst. J.*, vol. 13, no. 2, 2019.

[19] M. F. Hossain, A. U. Mahin, T. Debnath, F. B. Mosharrof, and K. Z. Islam, "Recent research in cloud radio access network (c-ran) for 5g cellular systems - a survey," *Journal of Network and Computer Applications*, vol. 139, 2019.

[20] A. M. Ahmed, S. A. Hasan, and S. A. Majeed, "5g mobile systems, challenges and technologies: A survey," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 11, 2019.

[21] I. Lopez, M. Aguado, C. Pinedo, and E. Jacob, "Scada systems in the railway domain: Enhancing reliability through redundant multipathtcp," in *Proc. of the IEEE Conf. on Intelligent Transportation Systems*, 2015.

[22] M. Alaluna, L. Ferrolho, J. R. Figueira, N. Neves, and F. M. V. Ramos, "Secure virtual network embedding in a multi-cloud environment," *CoRR*, vol. abs/1703.01313, 2017.

[23] E. G. N. . V1.1.1, "Network Functions Virtualisation (NFV); Terminology," *IEEE Network*, vol. 1, no. 5, pp. 1–50, 2013.

[24] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, 2008.

[25] X. Cheng, S. Su, Z. Zhang, K. Shuang, F. Yang, Y. Luo, and J. Wang, "Virtual network embedding through topology awareness and optimization," *Computer Networks*, vol. 56, no. 6, 2012.

[26] G. Marchetto, R. Sisto, F. Valenza, and J. Yusupov, "A framework for verification-oriented user-friendly network function modeling," *IEEE Access*, vol. 7, pp. 99 349–99 359, 2019.

[27] A. Khurshid, W. Zhou, M. Caesar, and P. B. Godfrey, "Veriflow: Verifying network-wide invariants in real time," in *Proc. of the ACM 1st Workshop on Hot Topics in Software Defined Networks*, 2012.

[28] A. Fiessler, C. Lorenz, S. Hager, B. Scheuermann, and A. W. Moore, "Hypafilter+: Enhanced hybrid packet filtering using hardware assisted classification and header space analysis," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, 2017.

[29] A. Volgenant, "Solving some lexicographic multi-objective combinatorial problems," *Europ. J. of Operational Research*, vol. 139, no. 3, 2002.

[30] J. Marques-Silva, J. Argelich, A. Graça, and I. Lynce, "Boolean lexicographic optimization: algorithms & applications," *Annals of Mathematics and Artificial Intelligence*, vol. 62, no. 3, 2011.

[31] South African National Research Network (SANReN). Accessed on 2018. [Online]. Available: https://www.sanren.ac.za/south-african-nren/

[32] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying nfv and sdn to lte mobile core gateways, the functions placement problem," in *Proc. of the 4th ACM Workshop on All Things Cellular: Operations, Applications, & Challenges*, 2014, pp. 33–38.

[33] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," in *Proc. of the 3rd Int. Network Optimization Conf.*, 2007.

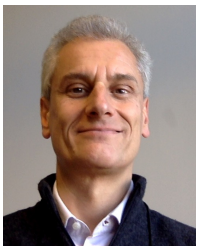
[34] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proc. of the 10th ACM SIGCOMM Conf. on Internet Measurement*, 2010.

[35] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with rocketfuel," *IEEE/ACM Trans. Netw.*, vol. 12 (1), 2004.

[36] A. Abu-Elanien, M. Salama, and K. Shaban, "Modern network reconfiguration techniques for service restoration in distribution systems: A step to a smarter grid," *Alexandria Engineering J.*, vol. 57, no. 4, 2018.

[37] M. Peuster, M. Marchetti, G. G. de Blas, and H. Karl, "Automated testing of nfv orchestrators against carrier-grade multi-pop scenarios using emulation-based smoke testing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, 2019.

Guido Marchetto received the Ph.D. degree in computer engineering from the Politecnico di Torino, in 2008, where he is currently an Associate Professor with the Department of Control and Computer Engineering. His research topics cover distributed systems and formal verification of systems and protocols. His interests also include network protocols and network architectures.



Riccardo Sisto received the Ph.D. degree in computer engineering from the Politecnico di Torino, Italy, in 1992. Since 2004, he has been a Full Professor of computer engineering with the Politecnico di Torino. He has authored and coauthored more than 100 scientific papers. His main research interests include formal methods, applied to distributed software and communication protocol engineering, distributed systems, and computer security. He is a Senior Member of the ACM.

Fulvio Valenza received the M.Sc. (summa cum laude) in 2013 and the Ph.D. (summa cum laude) in Computer Engineering in 2017 from the Politecnico di Torino, Torino, Italy. His research activity focus on network security policies. Currently he is a Researcher at the Politecnico Torino, Italy, where he works on orchestration and management of network security functions in the context of SDN/NFV-based networks.

Jalolliddin Yusupov received the M.S. degree in computer engineering from the Politecnico di Torino, Italy, in 2016, where he is currently pursuing the Ph.D. degree in control and computer engineering. His primary research interests include formal verification of security policies in automated network orchestration. His other research interests include modeling, cyber physical systems, and cloud computing systems.

Adlen Ksentini received the Ph.D. degree in computer science from the University of Cergy-Pontoise on QoS provisioning in IEEE 802.11-based networks. Since 2106, he has been a professor with the Communication Systems Department of EURECOM. His current research topics lie in the field of architectural enhancements to mobile core networks, mobile cloud networking, network functions virtualization and software defined networking.