Telecommunication Systems (2019) 71:505–515
https://doi.org/10.1007/s11235-018-0523-4

Enhancing network resources utilization and resiliency in multi-domain bandwidth on demand service provisioning using SDN

Alaitz Mendiola[1] · Jasone Astorga[1] · Eduardo Jacob[1] · Kostas Stamos[2]

Published online: 30 October 2018

Abstract

This paper presents a solution to increase bandwidth utilization and to reduce the time necessary to recover from network failures in multi-domain bandwidth on demand service provisioning. The proposed solution is based on software-defined networking (SDN) in order to implement novel traffic engineering (TE) techniques. In fact, most standards development organizations and researchers agree that the logically centralized control plane and high programmability of SDN can revolutionize TE with novel and powerful strategies. In this context, the dynamic path computation (DynPaC) framework, relies on the OpenFlow SDN protocol to provide resilient layer 2 services with bandwidth guarantees within OpenFlow domains. In order to make DynPaC support multi-domain service provisioning, it has been extended with an additional operational mode and REST API calls, making it fully compliant with the Network Services Interface-Connection Service protocol. In this way, DynPaC can behave as a network resource manager (NRM) in the network services framework. As a result, the presented solution is the first OpenFlow-enabled NRM with resiliency capabilities that implements novel TE strategies that enhance the utilization of network resources. In order to validate the proposal, it has been implemented using the ONOS network operating systems and several tests have been conducted using diverse service demand patterns. The obtained results show that DynPaC is suitable to be used as an NRM and that the number of accepted services increases thanks to the flow reallocation mechanism included in the framework, achieving in this way a better utilization of network resources.

Keywords Software-defined networking · Bandwidth on demand · Traffic engineering · Multi-domain connectivity

1 Introduction

In this new era of large-scale particle physics or genomic experiments, the e-science is at its peak. Researchers and innovators require novel networking services to access large amounts of data stored in remote data centers, so as to distribute the data obtained in their experiments to be processed in high performance computing facilities. Likewise, big data is revolutionizing industries and businesses, generating new service demands that require disruptive traffic engineering (TE) strategies.

To satisfy this sort of requirements, Internet Service Providers (ISP) and National Research and Education Networks (NREN) have started to include novel types of services in their portfolios, like bandwidth on demand (BoD). In a nutshell, BoD allows end-users to request high capacity, short term and flexible circuits with bandwidth guarantees using an automated provisioning tool, and it has become a very popular service among companies and researchers. When providing BoD, TE strategies are essential to satisfy

This work was supported in part by the EC through the Horizon 2020 Research and Innovation Programme (GN4) under Grant 731122 and in part by the Spanish Ministry of Economy, Industry and Competitiveness through the State Secretariat for Research, Development and Innovation under the "Adaptive Management of 5G Services to Support Critical Events in Cities (5G-City)" Project TEC2016-76795-C6-5-R.

✉ Jasone Astorga
jasone.astorga@ehu.eus

Alaitz Mendiola
alaitz.mendiola@ehu.eus

Eduardo Jacob
eduardo.jacob@ehu.eus

Kostas Stamos
stamos@cti.gr

[1] Department of Communications Engineering, University of the Basque Country UPV/EHU, Bilbao, Spain

[2] Technological Educational Institute of Western Greece, Patras, Greece

the Service Level Agreements (SLA) subscribed with the users, because once a service request is accepted by the provisioning tool, the resources that are or will be consumed by the service are reserved. This characteristic of BoD and advanced reservation systems in general can lead to resource over-provisioning, where the TE mechanisms to optimize network resources' utilization become of uttermost importance.

Since the appearance of software-defined networking (SDN) [1], the research community and the Standards Development Organizations have agreed on the potential of this new networking paradigm to revolutionize TE. As stated by the Open Networking Forum, SDN is a disruptive networking paradigm that proposes the decoupling of the forwarding and control planes, and the use of open interfaces to program the network devices from a logically centralized control plane. This approach can benefit network operators by enabling the automation of service provisioning and by reducing the network operational cost. Additionally, it also makes possible to apply novel and disruptive TE strategies to optimize network resources' utilization based on exploiting the finer granularity of the data planes, or even include efficient resiliency mechanisms thanks to the logically centralized control plane aware of the complete network state. Consequently, many NRENs and big telecommunication companies such as AT&T have started to work on novel services powered by SDN, aware of the benefits provided by technologies such as OpenFlow [2], NETCONF [3] or ALTO [4].

More specifically, the pan-European Research and Education Network GÉANT has started to work on the SDN-ization of its services, including the BoD service, which is currently provided through the AutoBAHN provisioning tool [5]. One of the most interesting features of GÉANT's BoD is that it enables the provisioning of multi-domain dynamic circuits, which is achieved through the Network Services Interface-Connection Service (NSI-CS) protocol [6], a technology agnostic protocol part of the network services framework (NSF) [7] that allows to deploy services across multiple and heterogeneous domains. Moreover, given the nature of current research initiatives and projects, where researchers distributed all around the world must cooperate, the future GÉANT's SDN-based BoD service must be able to keep guaranteeing its interoperability with the BoD provisioning tools of other organizations.

With this scenario in mind, this paper presents a solution for an SDN-based BoD service with multi-domain capabilities. It is based on the utilization of the dynamic path computation framework (DynPaC) [8], a powerful advanced bandwidth reservation system for SDN that provides resilient VLAN-based L2 services taking into consideration bandwidth constraints in OpenFlow domains. In order to be able to interoperate with the BoD services of other organizations, regardless of the transport technology being used, DynPaC has been designed to be NSI-CS compliant, acting as the network resource manager (NRM) of OpenFlow domains. This strategy will allow GÉANT to easily migrate its network resources to SDN-enabled ones, while keeping the strategic connections to provide worldwide connectivity services with other NSI-CS-capable domains, like the American Internet2 or the Energy Sciences Network (EsNET). Moreover, the future SDN-based BoD will benefit GÉANT by reducing the capital expenditures (CAPEX) and operating expenses (OPEX), by minimizing the currently excessive manual configuration (which is error prone) and by optimizing the network resources' utilization. Additionally, the DynPaC framework is currently implemented as an application for the open network operating system (ONOS) [9], which is a very powerful, modular and flexible SDN framework with multiple southbound interfaces, including support for OpenFlow 1.3. This provides also the possibility to further extend DynPaC, and therefore the SDN-based BoD service in GÉANT, not only to support OpenFlow domains, but also other SDN technologies.

The rest of the paper is structured as follows. Section 2 reviews current proposals for multi-domain SDN-based BoD solutions. Section 3 briefly describes the architecture of the DynPaC framework and the features that it provides. Then, Sect. 4 presents the REST interface that has been designed to support the integration of the DynPaC framework as an NRM on the NSF. Finally, Sect. 5 provides a performance evaluation of the implementation, while Sect. 6 summarizes the conclusions and presents the future work.

2 Related work

As mentioned before, BoD is a service provided by multiple NRENs and ISPs, and some of them have already started to integrate their solutions with SDN technologies. One of the most representative alternatives is the one proposed by the OLiMPS project, where they extend the on-demand secure circuits and advance reservation system (OSCARS) to operate over SDN domains [10]. OSCARS is the software framework used at EsNET to provide BoD circuits, and it is based on the path computation element (PCE)-based architecture [11]. Originally limited to MPLS/GMPLS domains and without an automated network topology discovery mechanism, the OLiMPS project enhanced the OSCARS software framework to overcome such limitations and support OpenFlow domains, first using the FloodLight controller and later using the OpenDaylight network operating system [12]. In addition, OSCARS provides multi-domain capabilities through the inter-domain controller protocol first and with NSI-CS later, as it happened with GÉANT's AutoBAHN.

Other NRENs are also trying to enhance their BoD systems with SDN to evolve from static to dynamic layer 2 circuit provisioning. This is the case of Internet2, which currently provides static BoD among its Advanced Layer 2 services [13]. In addition, through the open exchange software switch they provide dynamic layer 2 virtual circuits to users, which is based on OpenFlow 1.0. This service provides very interesting features, such as automated switch and topology discovery, VLAN usage monitoring or NSI-CS support for multi-domain circuit provisioning. However, it lacks the mechanisms to enforce quality of service (QoS) at the network devices. In order to solve this limitation and provide full SDN-based BoD, they are currently updating their system to support OpenFlow 1.3.

In general, PCEs are becoming increasingly popular not only to provide BoD services, but all kinds of services that require advanced TE strategies. For instance, several solutions for WANs using SDN include a PCE in their architecture, which makes possible to optimize network resources' utilization. An example of such solutions is the Cisco WAN Automation Engine [14], a tool able to provide optimized services like BoD, premium routing and global load balancing. Similarly, Juniper Networks' NorthStar Controller [15] relies on a stateful PCE to achieve a granular visibility and control of IP/MPLS flows in large networks with the aim of optimizing the use of network infrastructures while taking into account specific user-defined constraints. Analogous approaches are also followed by some SDN-based solutions for the optimization of the inter-data center WANs. On the one hand, Google uses an SDN-based TE solution to optimize the link utilization in their B4 network, which relies on a fair flow allocation mechanism that uses statistical information about the network usage [16]. On the other hand, Microsoft Research proposes SWAN [17], a system able to improve network resources' utilization based on the coordination of the services' sending rates and a centralized path allocation mechanism. Their TE strategy is engineered to leverage the high granularity available at the OpenFlow data plane, making use of traffic reallocation and disaggregation mechanisms to increase the amount of traffic carried out by the network. However, these solutions do not provide multi-domain capabilities.

All in all, as far as the authors know, our proposal is the first OpenFlow 1.3 enabled NRM with resiliency capabilities that enforces the required QoS at the network devices to guarantee the SLAs subscribed with the users. Furthermore, most SDN-based solutions perform path computation using PCEs to apply constrained-shortest path first (CSPF) algorithms, while the DynPaC framework implements a custom algorithm with support for flow reallocation to accept new service demands that otherwise would not be accepted.

3 DynPaC, a dynamic path computation framework for SDN

Currently, the SDN ecosystem consists of a plethora of technologies, ranging from pure management protocols such as OVSDB or OF-CONFIG to more complex frameworks and technologies such as OpenFlow or ForCES, which define new and more powerful forwarding planes and protocols [18]. Having this heterogeneity in mind, this section presents a reference architecture to support BoD service provisioning in software-defined networks, DynPaC. It defines a set of modules and features that allow to optimize network resources' utilization while providing resilient L2 circuits with QoS guarantees.

As depicted in Fig. 1, the DynPaC framework consists of seven different modules and a REST API, which will be further explained in Sect. 4. It follows a modular approach to ease the introduction of new modules and functionalities and to support the rapid modification of the existing ones.

- *Service manager* the core module of the entire architecture. It is in charge of the coordination among the remaining modules and storing the information about previous reservations.
- *PCE* the module in charge of the path computation for the requested services.
- *Topology abstraction* the module in charge of abstracting the topology to ease the resource management and the path computation.
- *Resiliency* the module in charge of listening to topology events and reacting upon link failures and network updates.
- *Network programmer* the module in charge of programming the network devices to install or remove flows according to the service's lifecycle.
- *GUI* a graphic user interface that allows users to request the services using a web browser.
- *CLI* an interface that allows users to request the services from the ONOS CLI console.
- *REST API* an open API that allows to interact with external entities.

The current implementation of the DynPaC framework is based on ONOS, and it is meant to operate over OpenFlow enabled domains. As mentioned before, the reference architecture establishes the functionalities that each module must implement, but it does not specify how this must be achieved. This allows to easily exchange current modules, as long as they provide the required functionalities. The following subsections provide a detailed overview of how the current implementation of the DynPaC framework works.

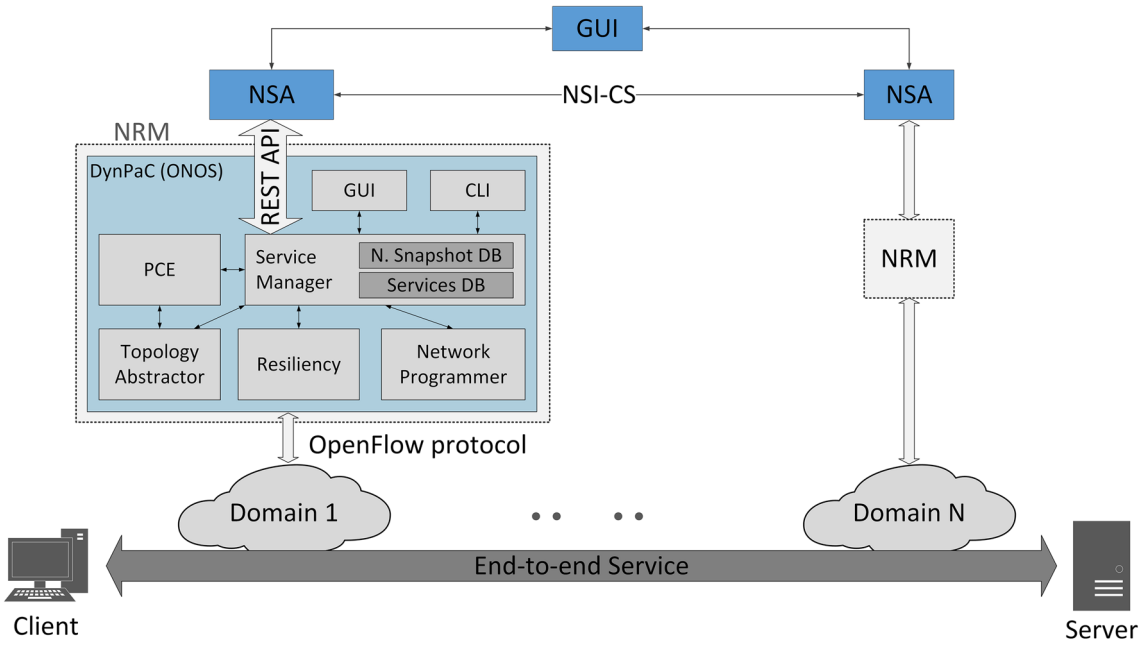


Fig. 1 Integration of the DynPaC framework as NRM for multi-domain BoD service provisioning

3.1 Network topology discovery

When the DynPaC application is activated in the ONOS network operating system, it first retrieves the topological information via de topology abstraction module. This information is later used for the correct path computation.

DynPaC relies on the topology service, the edge port service and the device service provided by ONOS to discover and abstract the topology. To ease the path computation between two egress points, DynPaC assumes bidirectional and symmetrical services. This assumption makes possible to abstract the topology using a vector, in which each position refers to a bidirectional link and its value to the maximum allowed bandwidth as advertized by the OpenFlow devices.

DynPaC also assumes that the services can only be provided between two endpoints within a domain. Therefore, in order to build the list of user-selectable (source or destination) endpoints, the topology abstraction module identifies among all the nodes inside the domain the edge-nodes; and among their ports, it selects the ports that are not used to connect to other nodes, that is, that are facing users.

3.2 Path pre-computation

Once the topology is retrieved, the DynPaC framework (via the PCE module) pre-computes all the possible paths between the endpoints (as identified by the topology abstraction module), that is, the networking devices facing end users or external administrative domains. The list of possible paths is then stored in a non-persistent database, where the paths are ordered prioritizing the shortest paths. This approach allows to accelerate the processing of a new service request.

3.3 Accessing the DynPaC GUI

In order to support BoD service requests within a domain, the DynPaC framework extends the ONOS GUI to provide a user-friendly tool to interact with the system. The DynPaC GUI is a single web page that can be opened from the ONOS GUI menu, based on ONOS' AngularJS building blocks for its layout and JSON communication with the ONOS/DynPaC back-end. It provides detailed and real time information about the already accepted services in the network, including information about their state, whether they are active or reserved, the VLAN assigned to the service, etc.

It also provides an easy way to request new services by specifying parameters such as the requested VLAN ID or VLAN range, the start and end times of the service, the source and destination endpoints, the requested bandwidth and the type of service. Feedback about the success or not of the request is then provided by the DynPaC service manager.

3.4 New service request

DynPaC provides support for two types of BoD services, depending on whether they have a totally disjoint backup path guaranteed or not. On the one hand, gold services are guaranteed with a backup path, which is pre-computed and automatically installed in the network in case of link failure. When a user requests a gold service, the system analyzes if

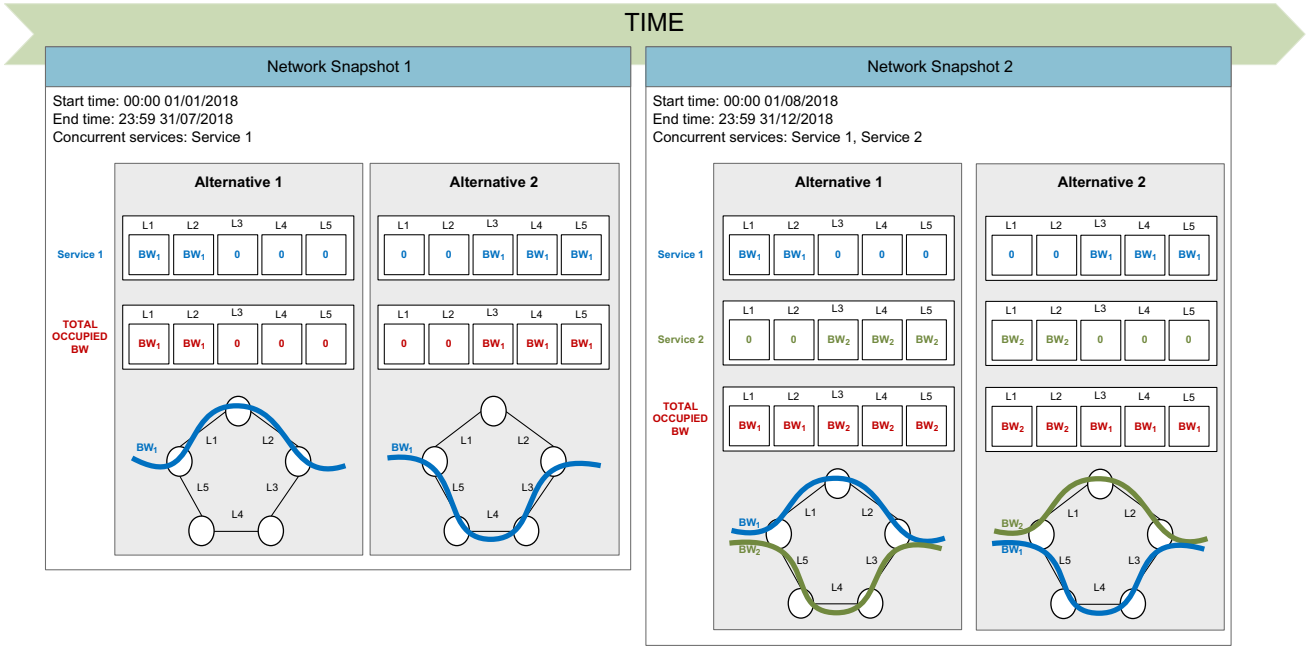


Fig. 2 Network snapshot's structure and transition in time

there are enough resources to provide both the primary and the backup path. If there are not enough resources available for the backup path, the system notifies the user that it is not able to provide the requested service. On the other hand, regular services are not guaranteed with a backup path. However, the architecture does provide a backup path for these types of services when there are enough resources available. The difference is that if there are not resources available for the backup path, the service request is processed normally, and the user gets notified that the service request has been accepted but without guarantees.

DynPaC allows to reserve time-bounded resources for future BoD services. Therefore, it is a framework with resource reservation capabilities and scheduling functionalities. In order to handle the time-dependent network state, the service manager implements network snapshots, which in the DynPaC framework are used to represent the state of the network in a specific period of time. Network snapshots provide information about the concurrent services at a given time interval and the different path alternatives that are viable to guarantee the provisioning of the already accepted services. Each path is represented with a vector, where each position represents the bandwidth consumed by the service in a given link. The service manager uses the list of all the possible paths for a given pair of endpoints provided by the PCE to generate the different alternatives, which are the path combinations that guarantee that there are enough available resources in the network to allocate all the concurrent services in the snapshot. Figure 2 represents the evolution of the network resources' utilization using the network snapshots. In particular, it depicts the different alternatives that exist to allocate the services in two consecutive network snapshots, the first one with a single concurrent service and the second one created as the result of an additional concurrent service reservation.

Upon a new BoD service request, the DynPaC framework executes an admission control procedure, consisting of two phases: *check* and *update*. The check phase starts when the service manager receives all the relevant information about the service, namely, the start and end time, the source and destination endpoints, the requested bandwidth and the type of service (gold or regular). With this information, the service manager analyzes if there are enough resources in the network (bandwidth, VLANs, etc.) to provide the requested service, taking into consideration all the snapshots it overlaps. This phase takes a time denoted as t_{check} and after its completion, the DynPaC framework notifies the user whether the requested service is accepted or not.

The second phase, update, only takes place for accepted services and it consists of selecting the optimal path for the requested service, as well as updating the necessary information in the network snapshot structures. In fact, the service manager implements an algorithm to select the shortest path among all the available paths that satisfy the connectivity requirements (for both the primary path and the backup, in case there is one), taking into consideration the bandwidth constraints and the previously reserved services. The shortest paths are retrieved from the list computed by the PCE module in the path pre-computation phase. Additionally, in order to avoid using always the same paths for the same source and

destination endpoints, paths of the same length are provided by the PCE to the service manager in a random fashion. In this way, a uniform distribution of services along the network is achieved. Note that randomization is only applied among paths of the same length, prioritizing always shortest paths.

It is worth noting at this point that DynPaC implements also a flow reallocation mechanism with the aim of maximizing the number of services accepted in the network, and thus, optimizing network resources' utilization. More specifically, if a new service cannot be accepted with the current distribution of ongoing or reserved services in any of the network snapshots it overlaps, DynPaC executes flow reallocation mechanisms independently in each of the affected network snapshots. Therefore, DynPaC analyzes the different path alternatives for the services that constitute the affected network snapshot and tries to reallocate services into alternative paths in order to make room for the newly requested service that otherwise could not be provided, thus, increasing the number of accepted services, also known as the service acceptance ratio. In the case of ongoing services, the new path is installed starting from the end, so that the whole path is already available when the first packets are transmitted through it, avoiding packet loss. On the other hand, in the case of services scheduled for the future, path reallocations can be performed without affecting the services' performance in any way.

The update phase, including optimal path selection, possible reallocation of flows and update of the information stored by the affected data structures, for all the overlapped network snapshots, takes a t_{update} time.

3.4.1 Reservation model

The time required to execute both phases, check and update is denoted as t_{setup}, as shown in Eq. 1. This is the minimum time that must elapse between consecutive service requests in order to guarantee that the data structures that represent the network state are up-to-date and stable.

$$t_{setup} = t_{check} + t_{update} \tag{1}$$

This t_{setup} time conditions how long, prior to the service's start time, must a service be requested to the DynPaC framework. This time is known as the *book-ahead interval* (t_{book_ahead}). That is, at least a t_{setup} time must elapse between the time at which a service is requested and the start time of this service. The relationship between t_{book_ahead} and t_{setup} is given in Eq. 2.

$$t_{setup} \leq t_{book\_ahead} \tag{2}$$

The relationship between all the previously defined time intervals is graphically depicted in Fig. 3.

On the other hand, the average t_{setup} time depends on whether a service request has been accepted in the network or rejected due to lack of resources, therefore, $t_{<setup>}$ is denoted as:

$$t_{<setup>} = t_{check} + p_A * t_{update} \tag{3}$$

being p_A the probability that a service is accepted in the network. This probability, is also known as the *Service Acceptance Ratio (SAR)* and it is a frequently used parameter to assess the performance of advance reservation systems. In fact, this parameter represents the amount of service reservation requests that are accepted by the system in comparison with the total amount of received service reservation requests, as shown in Eq. 4

$$SAR = \sum service_{accepted} / \sum service_{requested} \tag{4}$$

Tightly linked to the SAR, the *Blocking Ratio (BR)* represents the amount of service reservation requests rejected by the system, compared to the total amount of received service reservation requests:

$$BR = \sum service_{rejected} / \sum service_{requested} \tag{5}$$

3.5 Network devices programming

Once a service request is accepted by the service manager, it remains in reserved state until it's start time arrives. It is when the service's start time reaches that the service state changes to active, and the service manager informs the network programmer so that it installs the flows in the networking devices. The same happens when the service's end time reaches. In this case, the service is marked with the state finished, and the service manager informs the network programmer to remove the corresponding flow from the networking devices. The network programmer relies on the flow objective service provided by ONOS to generate the OpenFlow messages, and the application service to associate the generated flows to the DynPaC application. This module translates the information provided by the service manager to real OpenFlow control actions, which in the current implementation involves the usage of meters for rate limiting purposes and enforcing the QoS constraints.

As previously mentioned, of uttermost importance in the DynPaC framework is the smooth flow reallocation mechanism. DynPaC is able to reallocate the services into alternative paths in order to free the necessary resources for a new service request, but only when it is strictly necessary. This feature is handled by the service manager when a new network snapshot is about to become active, selecting the best alternative among the pre-computed ones in the network snapshot that guarantees minimum flow reallocations.

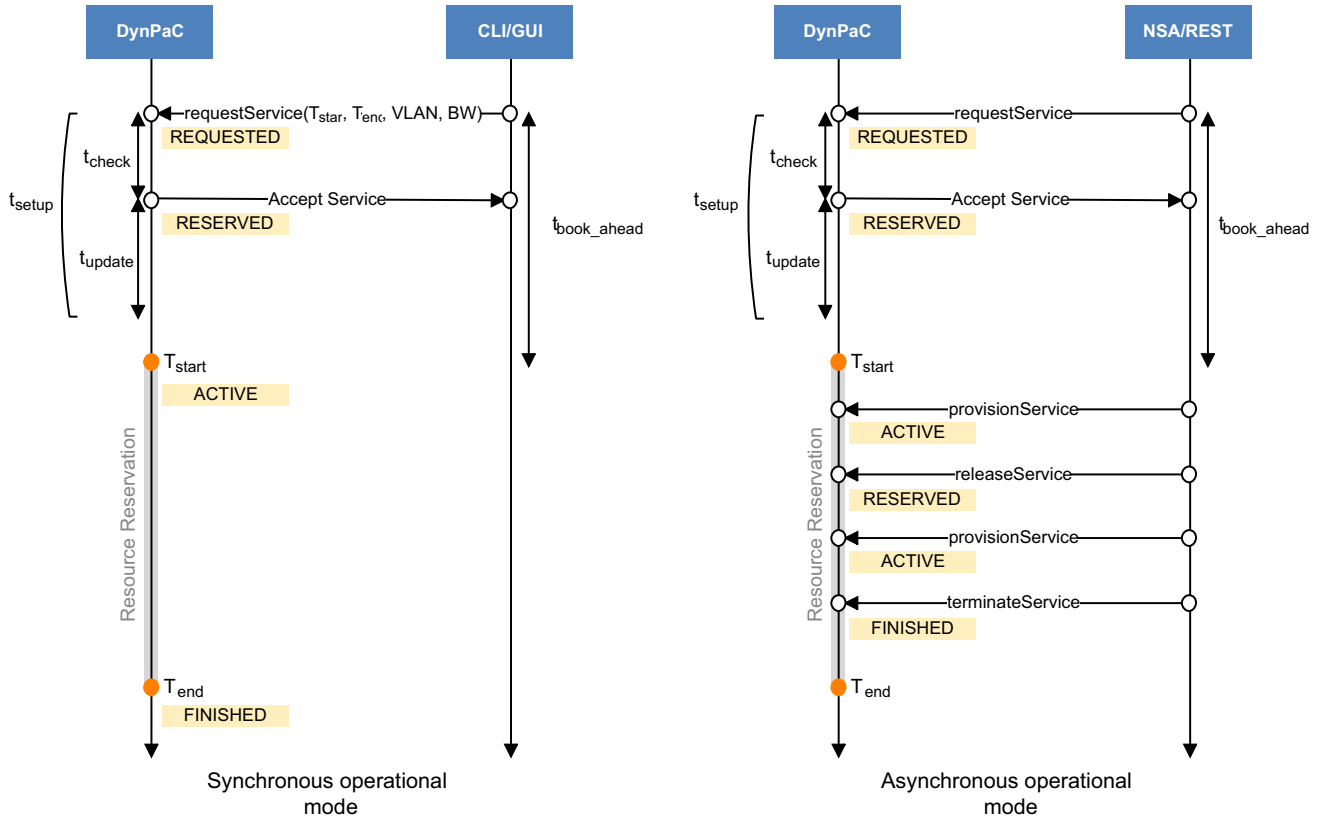


Fig. 3 DynPaC operational modes and services' lifecycle

In this way, the network devices are programmed only when it is necessary, avoiding unnecessary service disruptions due to switches' re-programming.

3.6 Reaction to link failures

One of the main functionalities provided by the architecture proposed in this paper is network resiliency. DynPaC has a resiliency module able to re-program the affected network devices to setup the flow entries associated to the backup path upon link failure. To be able to listen to link events, the resiliency module relies on the link listener service provided by ONOS. Upon reception of a link failure event, the resiliency module identifies the services that were being provided through that specific link, and triggers a procedure in the network programmer to install the new flow entries associated to the backup paths. It is worth reminding, that DynPaC differentiates gold and regular services. As a consequence, since the gold services have a guaranteed backup path, the resiliency module will always trigger the recovery procedure for these types of services. On the contrary, the regular services have a backup path only if there are enough resources in the network. Therefore, the resiliency mechanism needs to identify if the affected services are gold or regular.

4 NSI-CS support

As mentioned before, in order to provide multi-domain capabilities, the DynPaC framework has been designed to be NSI-CS compliant. This has been achieved extending the REST API exposed by DynPaC and introducing a second operational mode that allows DynPaC to act as the NRM of OpenFlow domains.

In NSI-CS, two main elements are involved in the end-to-end service provisioning. On the one hand, the network services agents (NSA) are in charge of handling the communication with the other NSA peers and of negotiating the end-to-end services' setup, activation/deactivation, modification and tear down. They also exchange topology information and thus, handle an abstracted form of the domain topology that is suitable for advertisement. In other words, the NSAs are in charge of the inter-domain aspects of the service provisioning. On the other hand, the NRM is the NSI component ultimately responsible for managing the transport resources. It thus needs to be able to model the detailed domain topology, maintain a resource utilization calendar and calculate intra-domain paths based on calendar and topology constraints.

In order to be able to use the DynPaC application as an NRM, the NSA needs to be able to map the internal NSI states within the DynPaC work-flow. NSI allows for

advance reservations, which is a concept already supported by DynPaC, but it also supports the concept of multiple provisioning/releasing cycles within a reservation, which is the process of building or tearing down the actual circuit without cancelling the booking of resources. This feature is very useful for the cases where there is a planned network downtime and the user needs to be aware of the exact times when the circuit is unavailable, without setting off monitoring alarms.

As a consequence, to be compliant with the NSI-CS work-flow, we have introduced an additional operational mode in the DynPaC framework, called *asynchronous network update*, in which the network devices are updated upon external request. This operational mode differs from the default mode at which the DynPaC framework operates, called *synchronous network update*, in which the network devices are updated using the start and end times specified in the service requests.

For the integration with external entities, the DynPaC framework exposes a REST API, which was originally designed to support the reservation or removal of generic L2 service demands. Furthermore, it also provides the means to retrieve information about the already reserved services and about the edge nodes with their corresponding ports and VLANs. In addition to these calls, we have extended the REST API in order to support the communication with the NSAs. More specifically, the new REST API calls have been designed to support the explicit service provisioning and the release of the resources of an active service. With these new API calls, the NSA is able to map the internal NSI states to the appropriate REST API calls so that the work-flow proceeds as specified by the NSI-CS standard [6]. Figure 3 depicts the two operational modes available in the DynPaC framework, and how the services' state varies in both cases upon reception of the new REST API calls.

5 Performance evaluation

In order to test the feasibility of the DynPaC framework as an NRM and as an advanced reservation system, we have implemented the proposed solution in a real-world testbed and we have performed a set of experiments in order to evaluate the performance of the platform from the users perspective. For that aim, we have conducted three types of experiments. First, we have measured the setup time, which is the required time to compute a path for a service request in order to guarantee that the DynPaC framework provides a response to the user in a time frame that can be considered real-time. Second, the blocking ratio, that is, the ratio of accepted service requests versus the rejected ones, and how it is improved by the flow reallocation mechanism. Finally, we have studied the effectiveness of the resiliency mechanism, by measuring the necessary time to install the backup paths of a set of services affected by a link failure. We have decided to evaluate these three metrics because they are related to the limitations present in the current BoD service that the DynPaC framework and the utilization of SDN solve.

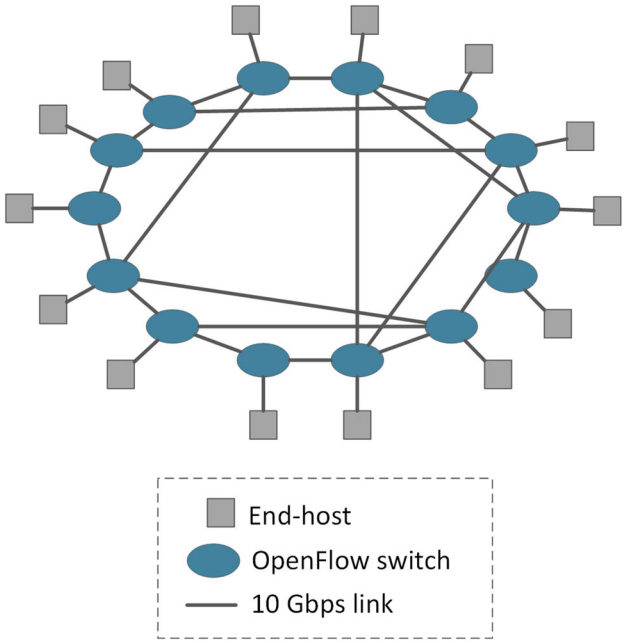
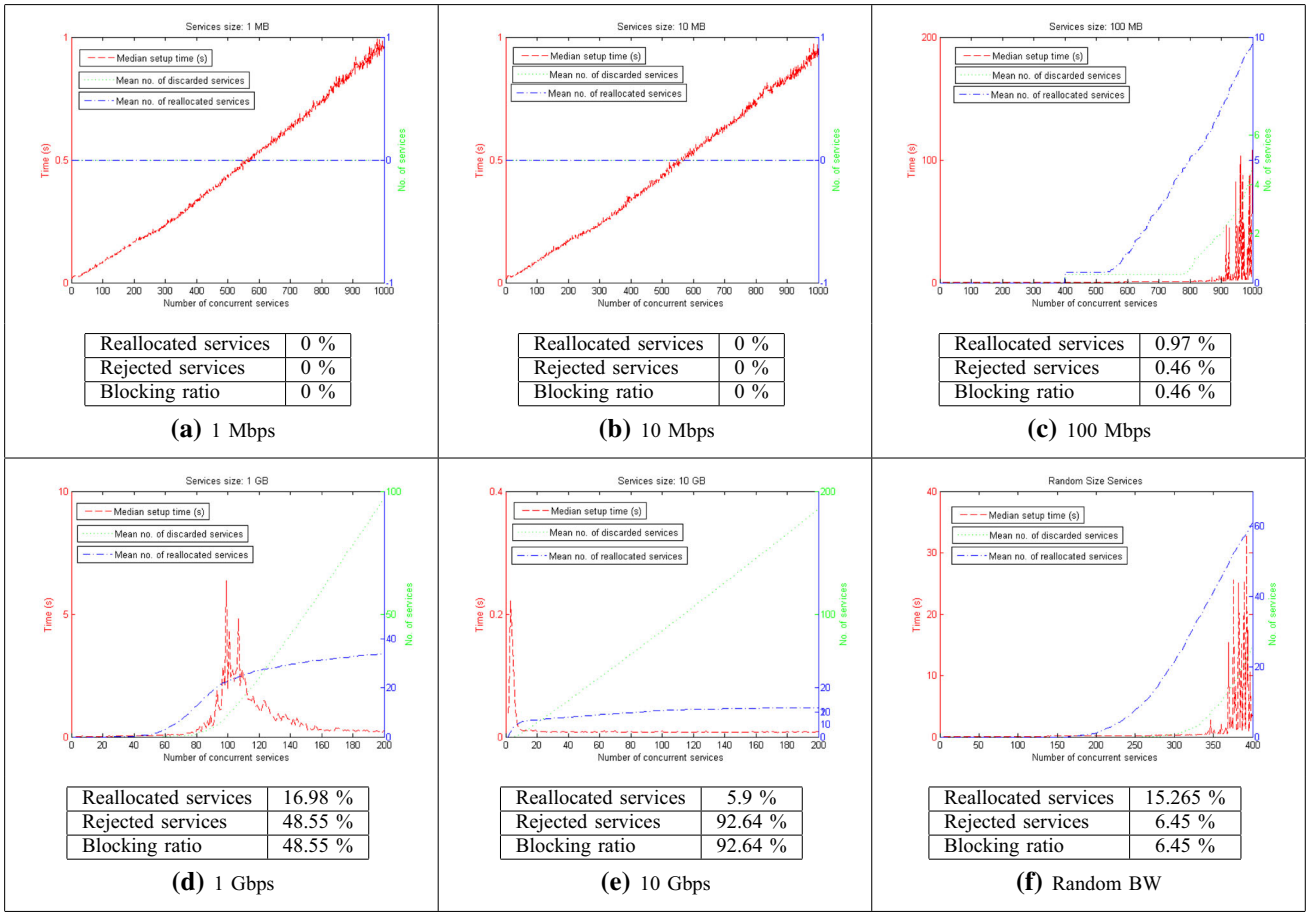


Fig. 4 Network topology (Color figure online)

The evaluation has been conducted using mininet to implement a network that consists of 14 Open vSwitches (OVS) with OpenFlow 1.3 support connected as depicted in Fig. 4. Specifically, the 14 OVSs are denoted as blue ovals in Fig. 4 and they are connected forming a ring, where each switch is directly linked to two adjacent neighbours. Nevertheless, in order to allow the existence of redundant paths, some additional links are also established between switches which are not adjacent neighbours in the ring. Additionally, each of the switches has an end-host connected, depicted as a grey square in Fig. 4. End-hosts are the only entities in the network that behave as traffic source or destinations. The reason for selecting this topology is that its complexity level is similar to the network used by GÉANT to provide BoD services. All the OVSs are controlled through an ONOS 1.4 instance with the DynPaC application installed. The whole setup runs in a Ubuntu 14.04 machine, with 12 GB of RAM, 40 GB of disk memory and four dedicated cores of an Intel i7 processor at 3.40 GHz. All the experiments have been repeated 30 times to consider a normal distribution.

5.1 Service setup time

This measurement refers to the time required to compute a path for a given service reservation. Figure 5 depicts the obtained results for 1 Mbps, 10 Mbps, 100 Mbps, 1 Gbps,



Reallocated services	0 %
Rejected services	0 %
Blocking ratio	0 %

(a) 1 Mbps

Reallocated services	0 %
Rejected services	0 %
Blocking ratio	0 %

(b) 10 Mbps

Reallocated services	0.97 %
Rejected services	0.46 %
Blocking ratio	0.46 %

(c) 100 Mbps

Reallocated services	16.98 %
Rejected services	48.55 %
Blocking ratio	48.55 %

(d) 1 Gbps

Reallocated services	5.9 %
Rejected services	92.64 %
Blocking ratio	92.64 %

(e) 10 Gbps

Reallocated services	15.265 %
Rejected services	6.45 %
Blocking ratio	6.45 %

(f) Random BW

Fig. 5 Setup time, number of reallocated and rejected services

10 Gbps and random bandwidth service requests from randomly selected pairs of source and destination nodes. We have limited the number of concurrent services in the three latest cases because the network was already saturated. The tests show that the time it takes to compute a path depends on the amount of already reserved services and the resources already consumed by them. The setup time gets increased until it reaches its maximum value when services are starting to be rejected and the number of possible alternatives to make future combinations gets reduced. We have decided to show the median value of the setup time because the average time gets affected by the setup time of the rejected services.

5.2 Blocking ratio

The blocking ratio refers to the percent of the service requests that cannot be satisfied due to some network resource unavailability. Figure 5 also shows the mean number of reallocated services and the mean number of rejected services for the same service requests distribution and a summary table with the percentage of reallocated services, rejected services, and the blocking ratios with and without reallocation mechanism.

The effect of the reallocation mechanism in the service setup time is minimal. For instance, in Fig. 5c the reallocation mechanism is used from service request number 400th without impacting the service setup time. However, the different service allocation alternatives that are maintained to make possible the flow reallocation does affect the time required to reject a service. All in all, even if the median setup time gets increased by the time associated to a service rejection, the blocking ratio gets improved, meaning that the path computation algorithm that runs in the DynPaC framework is improving the network resources' utilization.

5.3 Recovery time

Another fundamental feature of the DynPaC framework is that it is able to re-program the network devices to install a backup path upon link failure. As a consequence, we have conducted a third experiment to characterize the time required to restore all the affected services into their backup

path upon a link failure. For this particular experiment, we have selected a scenario in which a link failure affects all the services using that link, specifically, 100 gold services. Therefore, the backup path for all the affected services is already pre-computed and ready to be installed in the network. In such a scenario, we have measured the time required to restore all the affected services by installing their backup path. After repeating the experiment 30 times, the obtained results show that the average time necessary to have all the affected services running through their corresponding backup path is of 0.84 s, being the standard deviation of all the obtained measures of 0.29 s.

6 Conclusions

This paper presents DynPaC, a dynamic path computation framework able to provide BoD services in OpenFlow domains. It has been extended to be NSI-CS compliant, by incorporating a new operational mode that allows NSAs to control the life-cycle of the reserved services using a REST API.

In order to test the feasibility of DynPaC as an advanced reservation system able to provide BoD services, we have conducted three different experiments. First, we have demonstrated that DynPaC is able to compute a path for a given service demand in less than 1 s for different service sizes when the network is not overloaded. In this latter case, the time increases due to the application of the flow reallocation algorithm. However, this algorithm improves the overall Service Acceptance Ratio, achieving more than a 15% of improvement using random service demands, compared to when there is no flow reallocation. Finally, the resiliency mechanism requires around a second to reinstall the backup paths of 100 affected services upon link failure.

To the best of our knowledge, this is the first multi-domain SDN-based BoD system with resiliency capabilities. Additionally, being NSI-CS compliant, the DynPaC framework can be easily integrated with AutoBAHN, the current GÉANT BoD service, to support SDN domains. In this way, it will be possible to leverage the power and flexibility of modern SDN controllers without any change in the way it presents itself to BoD neighbours: the inter-domain protocol remains NSI and switching the current vendor-specific technology to DynPac/ONOS would be transparent to an external peer.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Open Networking Foundation (ONF). (2012). Software-defined networking: The new norm for networks. ONF White Paper. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf. Accessed September 11, 2018.
2. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., et al. (2008). OpenFlow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, *38*(2), 69–74. https://doi.org/10.1145/1355734.1355746.
3. Enns, R., Bjorklund, M., Schoenwaelder, J., & Bierman, A. (2011). Network configuration protocol (NETCONF). RFC 6241. Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc6241. Accessed September 11, 2018.
4. Seedorf, J., & Burger, E. (2009). Application-layer traffic optimization (ALTO) problem statement. RFC 5693. Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc5693. Accessed September 11, 2018.
5. Automated Bandwidth Allocation across Heterogeneous Networks (AutoBAHN). https://geant3.archive.geant.org/service/autobahn/pages/home.aspx. Accessed September 11, 2018.
6. Roberts, G., Kudoh, T., Monga, I., Sobieski, J., MacAuley, J., & Guok, C. (2014). NSI connection service v2.0. grid forum document (GFD). GFD-R-P.212. Open Grid Forum NSI-WG. https://www.ogf.org/documents/GFD.212.pdf. Accessed September 11, 2018.
7. Roberts, G., Kudoh, T., Monga, I., Sobieski, J., Guok, C., & MacAuley, J. (2014). Network services framework. Grid forum document (GFD), informational (I). GFD-I-213. Open Grid Forum NSI-WG. https://www.ogf.org/documents/GFD.213.pdf. Accessed September 11, 2018.
8. Mendiola, A., Astorga, J., Jacob, E., Higuero, M., Urtasun, A., & Fuentes, V. (2015). DynPaC: A path computation framework for SDN. In *Fourth European workshop on software defined networks*. https://doi.org/10.1109/EWSDN.2015.77.
9. Open network operating system. https://onosproject.org. Accessed September 11, 2018.
10. Guok, C., Robertson, D., Thompson, M., Lee, J., Tierney, B., & Johnston, W. (2006). Intra and interdomain circuit provisioning using the OSCARS reservation system. In *3rd international conference on broadband communications, networks and systems*. https://doi.org/10.1109/BROADNETS.2006.4374316.
11. Farrel, A., Vasseur, J.P., & Ash, J. (2006). A path computation element (PCE)-based architecture. RFC 4655. Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc4655. Accessed September 11, 2018.
12. Newman, H. B., Barczyk, A., & Bredel, M. (2014) OLiMPS: OpenFlow link-layer multipath switching. Final report. California Institute of Technology. https://www.osti.gov/servlets/purl/1163919. Accessed September 11, 2018.
13. Internet2's advanced layer 2 service. https://www.internet2.edu/products-services/advanced-networking/layer-2-services/. Accessed September 11, 2018.
14. Cisco WAN automation engine (WAE). https://www.cisco.com/c/en/us/products/routers/wan-automation-engine/index.html. Accessed September 11, 2018.
15. Juniper Networks NorthStar Controller. https://www.juniper.net/us/en/products-services/sdn/northstar-network-controller/. Accessed September 11, 2018.
16. Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., et al. (2013). B4: Experience with a globally-deployed software defined wan. *ACM SIGCOMM Computer Communication Review*, *43*(4), 3–14. https://doi.org/10.1145/2534169.2486019.

17. Hong, C. Y., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., et al. (2013). Achieving high utilization with software-driven WAN. *ACM SIGCOMM Computer Communication Review*, *43*(4), 15–26. https://doi.org/10.1145/2534169.2486012.
18. Mendiola, A., Astorga, J., Jacob, E., & Higuero, M. (2017). A survey on the contributions of software-defined networking to traffic engineering. *IEEE Communications Surveys & Tutorials*, *19*(2), 918–953. https://doi.org/10.1109/COMST.2016.2633579.

Alaitz Mendiola received her B.Sc., and M.Sc, degrees in Telecommunication Engineering in 2011 and her Ph.D, in 2017 from the University of the Basque Country (UPV/EHU). She worked as a researcher on the I2T research group from 2012 to 2017, participating in the EU funded GN3plus, DynPaC and GN4 projects. She joined GEANT in 2018 as an SDN developer. Her research interests include Traffic Engineering, Software Defined Networking and Network Function Virtualization.

Jasone Astorga received her B.Sc., and M.Sc. degrees in Telecommunication Engineering in 2004 and her PhD in 2013 from the University of the Basque Country (UPV/EHU). From 2004 to 2007 she worked at Nextel S. A., a Telecommunications enterprise. In 2007 she joined the UPV/EHU as a lecturer and as a researcher in the I2T research lab. Currently she is an assistant professor on the same Department. Her research interests include Software Defined Networking and Network Function Virtualization, IP-enabled wireless sensors, security in distributed environments and mobility management.

Eduardo Jacob received a B.Sc. in Industrial Engineering and a M.Sc. in Industrial Communications and Electronics from the University of the Basque Country (UPV/EHU) in 1991. He spent two years in a public R&D Telecommunications enterprise (currently Tecnalia). Later, he spent several years as IT director in the private sector before returning to the Faculty of Engineering of Bilbao and completing his Ph.D. in ICT in 2001. He is currently an assistant professor on that same Faculty, where he acted as Head of the Communications Engineering Department for five years, from 2012 to 2016. He also leads the I2T (Engineering and Research on Telematics) research lab. His research interests are related to applying Software Defined Networks to industrial communications, cybersecurity in distributed systems and IP-enabled wireless sensors.

Kostas Stamos received his Diploma, Master Degree and Ph.D. from the Computer Engineering and Informatics Department at the University of Patras. He has worked for the University of Patras, Research Unit 6 of CTI, for the Technical Educational Institute of Patras and GRNET, the Greek research network. His research interests include network applications, SDN, QoS, and bandwidth on demand.