

[Invited] FLATLANd:

A Novel SDN-based Flat Layer-2 Architecture Enabling NFV and Metro-Access Convergence

Frank Slyne, Marco Ruffini
 CONNECT Research Centre
 The University of Dublin, Trinity College
 Dublin, Ireland
 fslyne@tcd.ie, marco.ruffini@tcd.ie

Abstract— FLATLANd (Flat LAYer Two Large-scale Network) is a novel design for a Telecommunications architecture that is strictly flat and conducts traffic at layer 2 without the use of tunneling, VPNs nor labels. The architecture is inherently Open Access in that no one network nor service provider dominates over the others, as is the case in traditional wholesale and retail models for broadband access networks. Many of the layers and stacked components have been removed when compared to a traditional Telecommunications network. In this paper, we validate the functionality of the proposed architecture through the provisioning of basic and enhanced services. We compare the FLATLANd QoS framework against current practices. We present reduced protocol stacks, called TCPoE and UDPoE which are possible using the Layer 2 Wide Area addressing facilitated by FLATLANd. We conduct comparative simulations of classic and FLATLANd architectures using our custom-made NSIM network simulation package.

Index Terms— Software Defined Network, SDN, NFV, Metro-Access Convergence, Open Access, NSIM, TCPoE, UDPoE, Ethernet, layer 2.

I. INTRODUCTION

Traditional Telecommunications networks have evolved slowly over time, initially providing basic telephony services, then dial-up Internet access followed by high speed ADSL and VDSL, followed today by FTTCab and FTTHome. However, much of the architectural topology and network layers (Fig. 1) have remained unchanged, particularly in the access and metro portions of the network. An MPLS router tags the PPPoE tunnel with a Pseudo-Wire (PW) identifier and a Label-Switched Path (LSP) label. The PW is used to identify the path up to the Optical Line terminal (OLT). For each OLT different PWs identify different SPs and within an SP different service types (Video-on-Demand and VOIP). After the OLT, towards the Optical Network Unit (ONU), a VLAN tag, together with the MAC address, is used by the ONU to direct traffic through a pre-determined Traffic-Container (T-CONT) and GPON Encapsulation Method (GEM) port. In the case of PPPoE, there are significant Virtual B-RAS load and capacity constraints.

Tests done by BT [1] for example showed a maximum limit of 9,000 PPPoE sessions per virtual B-RAS.

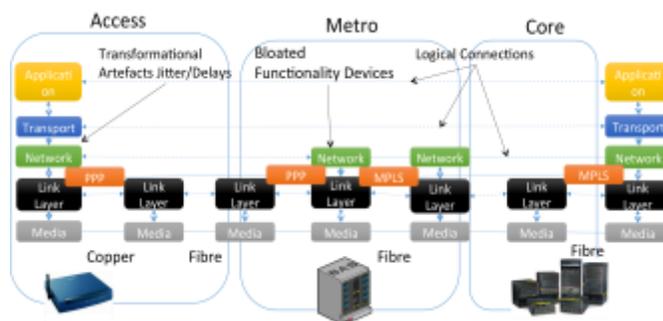


Fig. 1 Today's broadband telecommunications architecture

The tunneling and encapsulation for the transit of large connection volumes has significant downside such as restrictive network partitioning, slow reconfiguration times, and suboptimal dissociation between network platform and services. Each network layer and hop that is traversed has the potential to introduce artefacts such as jitter, excessive buffering and cross-layer authentication requirements. There are many locations in a TCP data path, where traffic may be buffered. These include network devices such as core and edge router nodes, broadband Remote Access Servers, customer premises equipment such as broadband routers and laptop network stacks as well as hosts within data centres. Buffers are judiciously placed at ingress ports to help absorb (without packet drops) any transient bursts of bandwidth that may occur on the traffic links. Logically, the size of a buffer [2] should be equal to the TCP congestion window which will vary with the Round Trip Time (RTT) of a TCP connection. Typical RTT [3] between sites within the same region is 20 msec, between sites on the same continent is 100 msec and between different continents 200 msec. The guideline [4] for the network equipment manufacturers is to provide buffers large enough to accommodate at least 250 ms worth of traffic passing through a device. For example, the 1 Gbps Ethernet interface on a router would require a buffer of 32 MB in size. If buffers are not adequately large [5] then TCP sessions with long Round Trip Time can experience excessive packet loss and TCP bandwidth reduction. However, due to the availability of inexpensive high density Dynamic RAM (DRAM), buffers are being made excessively large [6], and large buffers are being placed at

nodes along the path of traffic, such as CPE routers. TCP congestion avoidance algorithms rely on either packet Round Trip Times or packet drops to set the congestion window and the data throughput for a TCP connection. Where packets are buffered rather than dropped, the congestion algorithms do not alter their congestion windows appropriately. As a consequence, packets which have been subjected to long or variable buffering may arrive with either high latency or jitter [7]. The problem of continuously filled buffers which do not dissipate normally and function in a manner counter to their original purpose, that is, to improve Quality of Service, is called BufferBloat [8],[9]. Systems suffering from Bufferbloat have bad latency under load under some or all circumstances, depending on if and where the bottleneck in the communication's path exists. Bufferbloat encourages network congestion and destroys congestion avoidance in TCP transport protocols such as HTTP and BitTorrent. Network congestion-avoidance algorithms depend on timely packet drop. Unfortunately, bloated buffers invalidate this design presumption. Since it is quite common in the downstream network path for network elements to high-bandwidth ingress links and low-bandwidth egress link, the Bufferbloat problem is exacerbated [6] by traffic bursts on the high-bandwidth ingress links that can fill up the buffers without giving them a chance to be drained by the low-bandwidth egress links. For example, a buffer which is 1Mbyte in size takes 2 seconds to empty through a 4 Mbps pipe. This buffering effect affects UDP (that is non-TCP) applications as well, since application which require different mixes of latency and bandwidth, all share the same traffic links. Typical effects include skipped Mpeg Video synchronization frames, and time out of DNS resolver requests. In addition to specific events such as buffer saturation and Bufferbloat, network termination devices (such as CPE) need the ability to request and be granted a particular Quality of Service. Examples of Quality of Service requirements are the ability to maintain parameters such as jitter, delay or packet loss within give bounds.

In the next section, we outline the current approaches to dealing with these issues, through the use of queue management techniques and QoS Frameworks. Underpinning all QoS frameworks, whether they be legacy, existing or future, is the concept of the network flow. We proceed to define our solution to the overall legacy issues which we name FLATLANd, which is based on the widely known concepts of Network Function Virtualisation (NFV) and Software Defined Networks (SDN).

II. BACKGROUND

A. QoS Frameworks

QoS frameworks are typically categorized by how they deal with complexity, scalability and service differentiation. IntServ (Integrated Services) was one of the firstly proposed models for IP based QoS guarantees. IntServ (RFC 1633) [9] configures every router in a small network run by a single operator, where end users traffic patterns are predetermined. The Resource Reservation Protocol (RSVP) adapts the IntServ model for dynamic QoS provisioning of real-time interactive traffic over

larger and more complex networks [10]. The RSVP protocol uses signaling messages along the network path between sender and receiver. The down side of IntServ is that it is complicated and resource intensive [9]. In contrast to IntServ which deals with single flow instances, DiffServ (Differentiated Services) [11] reduces the volume of required flow state information in routers by dealing with flow aggregates [12]. Each edge device must set the appropriate DSCP bits based on the network's current QoS policy. DiffServ-enabled nodes are required to inspect the DSCP bits and respect the required QoS appropriate for that particular class of service. Overall, DiffServ [14] assurances are statistical in nature so there is no explicit alignment between the QoS requirement requested by an end application and the QoS delivered by the network. This makes DiffServ appropriate to networks with larger cores compared to IntServ.

In contrast to IntServ and DiffServ which make QoS decisions based on the IP packet header and are often criticised for increasing protocol management and overheads, MPLS makes QoS (and routing) decisions, based on short fixed length (shim) label in the packet header [13] [14]. When the label matches an entry in a routers forwarding table, the packet may be forwarded along an explicit Label Switched Paths (LSPs) [15]. MPLS configures an end-to-end path between routers and simplifies QoS classification and management [16]. However, every node along a network path must know what MPLS labels map to a particular class of service.

The Connectionless Approach overcomes the scalability issues of the IntServ model, by using an Automatic Quality of Service (AQS) mechanism instead of the RSVP protocol [17]. The AQS mechanism profiles the network traffic in real-time and defines the end-to-end QoS along the path of the traffic. The approach is scalable since it does not use signaling between nodes. However, in order to manage router bandwidth it retains the IntServ Model classifier, admission controller and scheduler. As a result, traffic handling capability is reduced because of the complex processing performed at each node. Connectionless approach is open to abuse by users that try to imitate other traffic types.

Dynamic Packet State (DPS) adopts the IntServ QoS admission control and scheduling and obviates the need for per-flow states in core routers [18]. The edge router inserts per-flow QoS classification into the IP packet header which must be read and updated by all (including the core) routers in the path of the traffic in real time. The architecture cannot be introduced gradually into the network, but must be done in a step change. Flow-Based Differentiated Services implement a flow based proportional QoS scheme based on three additional modules: a flow estimator for the number of active flows; a dynamic weighted fair queuing scheduler and a queue manager [11]. Flow-based DiffServ has the advantage of retaining the scalability features of the basic DiffServ, it also retains the disadvantages of having a limited number of Class of Services. Flow-Aware Networking (FAN) provides differentiation based on the current flow peak rate while protecting low-rate flows [19]. Admission Control maintains the quality of existing flows while restricting new flows (of all priorities) until network

congestion has improved. The functionalities for admission control and fair scheduling are implemented in a custom router called a Cross-Protect (XP) Router [20]. The XP Router does not require signaling between routers, the QoS calculation algorithms are lightweight so there is low processing overhead [21].

The Flow-State-Aware (FSA) Transport [22] mechanism uses DiffServ flow-aggregation in the Core and QoS mechanisms at the edge that are based on individual flows. While the Flow-state-aware transport technology is relatively similar to IntServ, it uses flow aggregations and is thus more scalable and less complex. FSA typically separates services from the underlying separating transport layer [23]. Flow-Aggregate-Based Services [24] enhances the FSA architecture and addresses three distinct types of congestion - instantaneous (packet-level) congestion, sustainable (flow-level) congestion and congestion avoidance. Instantaneous congestion is mitigated through the proper aggregation of flows and discard of packets. Sustainable congestion is resolved through rate limiting, and admission flow discards.

Overall, QoS frameworks are distinguished by whether they require signaling or not. Both Connectionless approach and FAN do not require any signaling and do not offer much service differentiation. IntServ, Flow-State Aware and Flow-Aggregated-Based Services outline in depth how to use signaling and offer greater differentiation, with numerous parameters to be assigned to each flow and multiple classes of service. Signaling is relatively complex limiting the scalability.

B. Traffic Flows

All flow-based QoS architectures, by their very nature, must understand the concept of a traffic flow as the entity that is subject to defined quality metrics. IETF defines a flow as a 5-tuple of source and destination addresses, source and destination ports, and the transport protocol (either TCP or UDP). In the IntServ QoS framework, traffic related to a single service is classified as a flow. Routers in the traffic path must apply the same pre-defined QoS rules to all packets within the flow. In the Connectionless QoS framework [25], a flow is defined as a stream of packets between client-server applications. In flow-based differentiated QoS frameworks, a flow is regarded as an aggregate of all transmissions between the same end users, defined by a unique pair of source and destination IP addresses that belong to the same class of service defined by a value of the DiffServ field (DS field) [26]. Unfortunately, when NAT obfuscates multiple sessions within a single flow, the source destination pair are identical. Similarly, NGN-based flow QoS frameworks, such as Flow State Aware transport and Flow-Aggregated-Based Services define a flow based on a unique pair of source and destination IP addresses that belong to the same class of service defined by a value of the DiffServ field or MPLS field.

C. Queue Management

Classic Active Queue Management (AQM) algorithms based on RED try to identify Bufferbloat by gauging how full buffers become. There are two problems with approaches based upon RED-based AQM algorithms [6]. Firstly, buffers may fill up

for legitimate reasons other than through Bufferbloat. Buffers may fill up due to short spurts of high volume traffic and then dissipate normally. These are called Good Buffers. Secondly, such algorithms do not facilitate remedial actions for TCP traffic streams buffered for long periods, as they do not discriminate based on the age of data in the buffer. Both CoDel [7] and PIE [27] try to preempt buffer saturation by either dropping packets or throttling high-bandwidth flows. They do this well in advance of the Saturated Tail algorithms. Similar to RED, PIE randomly drops a packet at the onset of the congestion, however, congestion detection is based on the queueing latency (similar to CoDel) unlike the queue length in conventional AQM schemes such as RED. PIE uses a combination of latency moving trends and whether latency is increasing or decreasing to determine the true levels of congestion. The CoDel (Controlled Delay) scheduling algorithm determines if a queue is good disciplined or bad based on the minimum age of packets in the queue. A good queue is where the minimum age of a packet is less than 5 milliseconds. For this algorithm to work, the timestamp of when the packet entered the queue must also be stored. When a packet is dequeued with an age greater than 5 milliseconds for a given window, the algorithm drops the packet. CoDel can infer buffer depths from the measured packet delays. The advantages of the CoDel algorithm are that the monitoring and the action of the algorithm require little processing overhead and require no additional configuration parameters. No action is taken against packets within a Good disciplined queue. The disadvantage of CoDel is that it requires changes to data structures within the queuing mechanisms of host and routing devices.

D. Software Defined Networks

The Open Network Foundation [28] defines Software Defined Network as a network “*architecture [that] decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services.*” SDN has the advantages of being “*dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today's applications.*” [29]. The concept of Software Defined Network (SDN) appears in different categories of networks ranging from carrier networks, data centres and central office networks through to home and wireless networks. SDN is relevant to physical, link, network and transport layers of the OSI and TCP/IP stacks, both individually but also in an amalgamation. EU FP7 project SPARC (Split Architecture) [30] demonstrated SDN in the Access and Aggregation network as well as a prototype of Network Function Virtualisation, through a Virtual Home Gateway and a Virtual BRAS. There are two (so-called) splits in the SPARC architecture. Firstly, there is the split between the Control and Data planes that allows the data and control planes to evolve separately from each other. The data plane extends reach, connectivity and bandwidth, while the control plane enhances service creation, control and delivery. Secondly, there is the split between the forwarding and processing elements. In a traditional telco network, these functions are distributed throughout the network, for example

at DSLAMs and customer home gateways with the result that these functions become isolated and degraded, though lack of manageability and enhancement. The split in forwarding and processing elements is familiar in the concept Network Function Virtualisation, where simplified forwarding components at the level of data plane are located in the field or remotely, with the processing elements concentrated in either data centre or central office environments.

The Application-Based Network Operations (ABNO) [31] is an SDN framework that uses MPLS and GMPLS multi-domain networks with PCE as the controlling agent and PCEP as the control protocol [29]. ABNO has a policy manager, an I2RS (Interface 2 Routing System) client, a Virtual Network Topology Manager (VNTM) for multi-layer co-ordination and an Application-Layer Traffic Optimisation Server. Southbound communication with components such as Openflow are achieved using a provisioning manager. Statefulness is provided by an LSP-DB and TED database. ABNO has been used in the IDEALIST project [32] to demonstrate the multi-domain and multilayer configuration of commercial equipment (such as ADVA, Juniper nodes and OTN 400 Gbps channels) and the validation of the PCEP extensions to support remote GMPLS LSP set-up.

ONOS [33] is an ONF [28] project with resources allocated to it by services providers such as AT&T and NTT, and research entities such as Internet2 and CREATE-NET. The objectives of the ONOS project are to provide a SDN platform with carrier-grade performance and availability, and forms part of the OpenCORD initiative. The ONOS project has outlined a number of use cases to demonstrate the carrier capability of the system. These are an SDN IP Peering use case, a Network Function Virtualisation as a Service (NFVaaS) use case and a use case demonstrating failover using IETF Segment Routing (Spring Project). The NFVaaS use case demonstrates a virtual OLT (vOLT) solution for GPON. ONOS does not rely solely on Openflow as its SDN control plane technology, as demonstrated in the Segment Routing use case. The PCE [34] use case looks at the issue of over-dimensioning of current Packet Optical cores so as to handle both network outages and peak bursts. ETSI promotes the standardisation for Fibre to the CAB (FTTcab), VDSL2 and G.Fast, and has looked at which traditional components may be virtualized [35]. These components include GPON OLT's, ONU's, DSL DSLAM and Broadband Remote Access Servers (B-RAS) and home gateway devices. The Broadband Forum has a number of working groups looking at SDN as part of Broadband (SD-313), Access Networks (WT-358) and as an enabler for Flexible Service Chaining (SD-326) and Network Function Virtualisation (WT-359) [36].

III. ARCHITECTURE AND FUNCTIONALITY

We propose the FLATLANd (Flat LAYer Two Large-scale Network) telecommunications network [37], which inherits the Portland [38] architecture for data centers (DC), as a paradigm to facilitate an efficient hierarchy of layer-2 switches and distributed Openflow tables (across ONU/OLT, electrical and

optical switches). The Portland architecture uses Ethernet MAC addresses for routing traffic within the DC, where an address allocation policy assigns pseudo-MAC addresses to each host and organizes the addresses into a hierarchical topology[37]. In the FLATLANd architecture we apply the same concept to telecommunications networks. Any network that uses Ethernet as a layer-2 protocol can potentially benefit from the FLATLANd architecture. Rather than preserving legacy devices such as B-RAS in their physical or virtual form as does SPARC [39], we re-architect the entire network from first principles. We target in particular next generation optical broadband networks, and take into consideration the convergence of access and metro networks, using the Long-Reach PON (LR-PON) architecture, discussed in [40], as a case study. Using protocols derived from XG-PON [41], LR-PON extends optical reach to customer ONU's to over 100Km, thereby bypassing the metro transmission network and enabling access-metro convergence through consolidation of central offices. The LR-PON allows for a larger split Ratio of 512, as opposed to 64 for XGPON [41]. In addition, due to the long feeder fibre distance, of around 100km, typically dual-homing protection mechanisms are envisaged [43],[44]. Typically, layer-2 Ethernet addresses of network devices and terminations are assigned during manufacturing and are thus uncorrelated to their location and other devices in their vicinity. This restricts their use in switched LAN and WAN segments, due to the impossibility to create any kind of hierarchical structure in the addressing scheme and forwarding tables. Through the use of pseudo-MAC addressing, the FLATLANd architecture overcomes such limitation by creating a structured Ethernet addressing domain that spans the entire network between the network terminations at the customer premise and the datacenter (Fig. 2), thus empowering wide area SDN at layer-2.

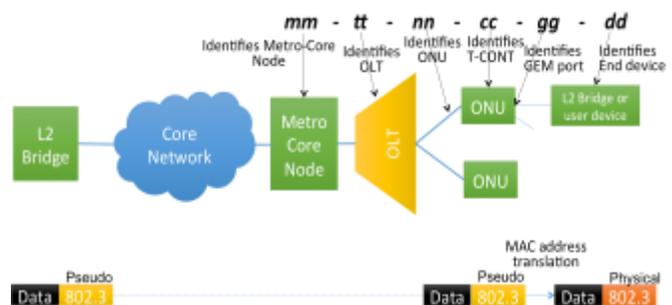


Fig. 2 FLATLANd FTTH function-level diagram

A translation is performed between the real (physical) address of the end device and the internal structured (pseudo) addressing used within the network. In the case of the PON, this translation is performed at the ONU GEM port. The mechanism partitions the internal 48-bit address space of an Ethernet layer into a number of arbitrary subfields, each routed to a different part of the network. The correlation between the real and pseudo addressing is performed dynamically by the SDN controller. For the LR-PON scenario we have identified a possible addressing scheme based on the following allocation: 'mm-tt-nn-cc-gg-dd'. Following the structure in Fig. 2, 'mm' identifies up to 4096 different metro-core nodes (12 bits), each

with up to 'tt' up to 4096 OLT ports (12 bits). It should be noted that in the LR-PON architecture [40] a Metro-Core (MC) node is a converged node terminating both access and core connections, thus combining functionalities of access, metro and core nodes. Within an OLT port, 'nn' identifies up to 4096 ONUs (12 bits), each with 16 'cc' T-CONTs (4 bits). A T-CONT is a group of logical connections that carries traffic within an ONU. Each T-CONT is identified by a unique Allocation Identifier (Alloc_ID) carrying traffic associated to one bandwidth type (i.e., QoS characteristic). The final 8 bits are split between GEM ports 'gg' (4 bits or 16 GEM ports per T-CONT) and devices 'dd' (4 bits or 16 devices per GEM ports). A GEM Port is a virtual port that encapsulates frames transmitted between the OLT and the ONU. Each traffic-class is assigned a different GEM Port. This would allow for example different users on the same ONU to acquire services from different providers concurrently. It should be noticed that we consider a classless address structure, where each block can have an arbitrary number of bits (up to a maximum sum of 48 bits, defined by the Ethernet address space limit). In FLATLAND, we see a flattening of layers within the access and metro portions of the network [37]. This is caused by some functions, such as B-RAS and PPPoE terminating modems being made redundant, and other network functions such as AAA services (Authentication, Authorization and Accounting) being virtualized at the periphery of the network, following an NFV approach. Secondly, there is potential for significant Capex and Opex improvements (reduced Operations and Maintenance) due to the adoption of white-boxes Openflow-based switches controlled by a unified SDN control plane. Thirdly, the network is inherently Open Access [42] in that the roles of infrastructure provider, network provider and service provider can be clearly demarcated. All devices are granted access to the network but subsequently may be dynamically or statically bound to the profile of a target service provider. Indeed the flexibility of the addressing scheme favors multi-tenancy, as parts of the address can be used for packet routing purposes and other parts for QoS and SP differentiation. Distinct flow rules are created for the metering of each class of traffic at each Metro-Core node, OLT and ONUs. These are separate from the rules necessary for forwarding flows. Traditional Service Providers have built dedicated networks so as to differentiate their services from other offerings. Differentiation has been based on factors such as availability and bandwidth. In the FLATLAND design, in order to allow sharing of a common infrastructure across all Service Providers, we propose mechanisms for bandwidth apportionment that are distributed throughout the network. Bandwidth apportionment may be performed at the root of the network, which has visibility of all traffic flows in the network, however, that would require a contiguous flow table which is unfeasibly large (with potentially 2^{48} entries). The FLATLAND architecture proposes two solutions to this issue, adopting distributed bandwidth apportionment: Geographical and per-Class. Geographical bandwidth apportionment applies control to the flows traversing each network element. For example, in order to apportion bandwidth according to a per-OLT basis, rules need to be applied at the upstream Metro Core

network. In order to apportion bandwidth on a Service Type or Service Provider basis in the Geographical model, rules need to be applied to the upstream TCONT and GEM ports. The existing flow rules can be modified with the meter tags on the output action. Per-Class bandwidth apportionment instead applies control to the flows traversing each network element. The key difference with the Geographical model is that distinct flow rules are created for the metering of each class of traffic at each Metro-Core, OLT and ONUs. These are separate from the rules necessary for forwarding flows. The advantage of per-Class bandwidth apportionment is that there is greater control over each Class of service across the network, whereas with Geographical, there is probably more efficient use of bandwidth. FLATLAND uses the broad and flexible Openflow definition of a flow, which has expanded from the basic 5-tuple to included other attributes such as MPLS labels, VLAN tags and IP TOS fields. All devices are granted access to the network but subsequently may be dynamically or statically bound to the profile of a target service provider. Indeed the flexibility of the addressing scheme favors multi-tenancy, as parts of the address can be used for packet routing purposes and other parts for QoS and Service Provider differentiation. Distinct flow rules are created for the metering of each class of traffic at each Metro-Core node, OLT and ONUs. These are separate from the rules necessary or forwarding flows.

FLATLAND is distinguished from IP-layer QoS frameworks such as IntServ, DiffServ [12] by providing QoS guarantees at layer 2. Indeed, FLATLAND can supplement IP-Layer QoS frameworks. Like Dynamic Packet State, FLATLAND has a stateless core and does not suffer from scalability restrictions of IntServ in the core. FLATLAND has inherent Admission Control since each flow must be specifically bound to a service. Like Flow-Aware Networks and the Connectionless approach, the basic FLATLAND model does not require end-user signaling, however, it can support signaling, if needs be, so as to provide service differentiation. This is demonstrated as part of the FLATLAND ingress admission control.

IV. FUNCTIONAL VALIDATION

We validated the operations of the SDN controller in the FLATLAND architecture in the stages required to register an ONU device with a given Service Provider: first pseudo address allocation and then layer-3 authorization. We validated the FLATLAND address partitioning and mapping scheme by replicating the 6-tier network hierarchy shown in Fig. 3 and test key NFV functionality such as service registration. Service registration is the process that allows each network element to obtain a pseudo-MAC address unequivocally associated to its physical MAC address, thus enabling its association to the FLATLAND network, and Layer-3 authentication. To validate the functionality on a virtual environment implemented on the Mininet platform [43]. A custom Openflow controller was developed, derived from the base POX implementation, and extended with memory-based Redis database. The Redis database is low-latency and can be (geographically) distributed across many physical machines, with some implementations

handling millions of queries per second. Redis preserves transactionality between nodes.

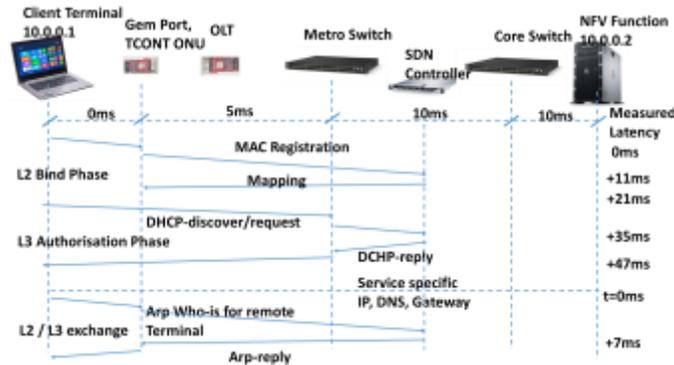


Fig. 3 Service Registration on SDN/NFV testbed

While there is a single master read/write node, changes in this database can be instantaneously mirrored across many read only nodes. For the current purposes, the database maintains the mappings between all real-mac, pseudo-mac addresses, IP addresses, and flows both in the network in the datacenter for the virtualization of Network Functions. Fig. 3 shows the emulated network architecture, inclusive of emulated latency times between the network elements (the values used are only indicative of the LR-PON case study). The test initiates with the Layer-2 Bind Phase, where the client device at the GEM port of the ONU registers its interface on the network. This interface is configured to obtain its IP address from a DHCP server, situated centrally and upstream from the device.



Fig. 4 SDN Controller log of Service Registration

On sensing of a DHCP-discover/request packet, the layer-2 of the customer Openflow-enabled ONU, sends the DHCP packet to the centralized Openflow Controller [44]. At the ONU, the OpenFlow switching is operated by the ONU GEM port switch.. Since the Openflow Controller knows the ONU from which packets are received, the controller formulates a pseudo-MAC address appropriate to that ONU. The Openflow Controller then performs three actions which are recorded in the SDN Controller log in Fig. 4. The Openflow Controller database creates a forward and reverse mapping between the real- and pseudo-MAC addresses to allow fast database lookups. The mapping is then sent to the ONU as an Openflow rule. The layer-3 authorization phase is required for the ONU client to receive appropriate network layer facilities such as IP address, DNS settings and Gateway addressing. The system operates by the Openflow controller intercepting a DHCP-discover/request either as part of the layer-2 bind phase or as a retransmission of this request. The Openflow Controller constructs a DHCP-reply packet with the appropriate settings,

for transmission through the ONU switch, to the client. The Openflow Controller also constructs per-Service Provider IP addresses and DNS settings. In the ARP Exchange Phase, the end-points exchange IP addresses and MAC address pairings. Where the client device sends an IP packet to a data center, the ARP who-is request is broadcast upstream and the upstream device responds with an ARP response. The GEM Port switch performs a swap of real- and pseudo-MAC addresses for the client device. The metro switch intercepts the ARP who-is request destined for the pseudo-MAC of the client device. Finally, the controller performs a proxy-ARP functionality based on the pseudo-MAC address of the client device. Fig. 5 shows a wireshark packet trace of the service registration process from the perspective of the client terminal (10.0.0.1).

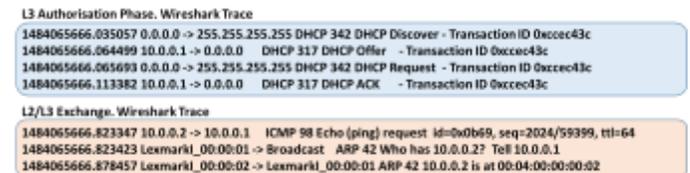


Fig. 5 Wireshark Trace of Service registration

In order to demonstrate the NFV functionality, a single layer datacenter switch was instantiated with small number of rules to direct the upstream and downstream traffic flows to and from the Network Function Virtual Machine. Linux LXC Container technology was used for the virtualisation, with the advantages that a low storage and processing overhead is imposed on the host environment. A basic traffic application was deployed on the Network Function Container, which both transited traffic between the ingress and egress interfaces as well as inspected and logged packet headers and payload. The testbed results show that registration times of around 30 ms can be achieved for a LR-PON based scenario. While such operations are generally not time-critical, these results demonstrate the type of benefits that a simplified SDN-driven flat architecture can bring about.

V. PERFORMANCE

We compared the performance of the FLATLANd and classic architectures using a Network simulator. Having reviewed NS3, we found the mandatory use of the CSMAhelper function for Ethernet interfaces and the full IP stack for network nodes too restrictive, so we developed our own simulator, which we call NSIM. NSIM is a network simulator that can simulate multi-layer fixed network functions such as hosts, Ethernet and Openflow switches, IP routers and Tunnelling routers. Its distributed architecture allows simulations to be run across multiple machines. Because it is written in Python, an open semi-interpretive language, other Python based toolsets such as *Numpy*, *Scipy*, *Scapy* and *Octave* can be used within the simulations. The fundamental NSIM building block of network links and network layers is the Duplex, which has two bi-directional buffered interfaces. The Duplex is configured to perform as a lossy or lossless component and its buffers may be configured with drop-tail, active queue (AQM) and aged queuing (CoDel) disciplines. The Datalink component is a subclass of the Duplex and adds additional transmission

characteristics such as latency and Bit Error Rate. Ethernet switches, IP routers and Vswitches are multi-layer Duplex components that switch at an Ethernet level, route at an IP level, and drop/pop MPLS, VLAN and PPP tags respectively. Currently, a rudimentary Openflow switch model is supported with a limited number of TCAM slots. UDP and TCP transport layers are supported, with a core subset of the TCP congestion control functions implemented. To benchmark the performance of the proposed Flatland scheme against the state of the art Classic QoS Frameworks, we simulated the models in NSIM and applied similar traffic profiles to the models. The classic architecture is modelled in Fig. 6 using a datacenter host1 connected to a network chain comprising of a core router, metro router, broadband access server, OLT, ONU, CPE and home router.

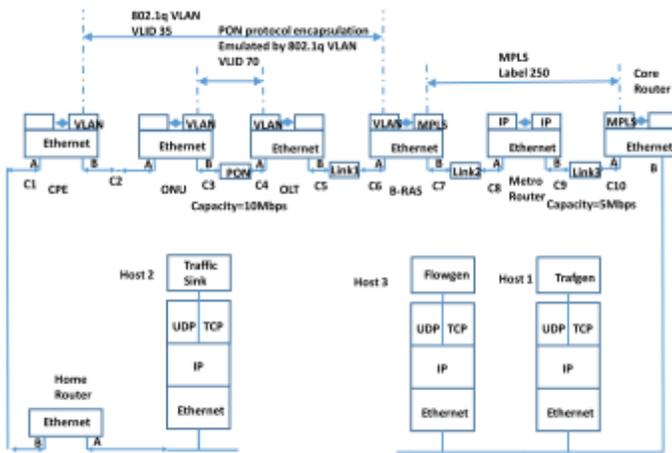


Fig. 6 Classic Architecture modelled in NSIM

The datacenter host (host1) application generates traffic, which is directed to a traffic sink (host2). All hosts are built with a UDP stack. Host2 is directed to drop traffic which is generated by the load traffic generator attached to host3. The target traffic application (trafgen) emits application packets at a rate of 1 packet every millisecond which is then encapsulated in UDP or TCP and the other lower levels in the protocol stack. All datalinks (pon, link1, link2 and link3) are configured with a Bit Error Rate (BER) of 10^{-12} . Links link1, link2 and link3 are not rate limited. The end-to-end latency between host1 and host 2 is 10 milliseconds. Traffic on the PON link are constrained by a rate limit of 10 Mbps. The traffic on link3 between the Core Router and the Metro Router is deliberately rate limited to 5 Mbps. Traffic between the ONU and the OLT is encapsulated in a 802.1q tunnel. Traffic traversing the ONU from the A to the B interface is tagged with VLAN ID 22. The VLAN tag on traffic going in the opposite direction (from interface B to interface A) is dropped. Traffic between the CPE and the BRAS is encapsulated in a 802.1q tunnel with Vlan ID 35. VLAN encapsulation is used to emulate PPPoE encapsulation.



Fig. 7 Applied Traffic

Traffic traversing the CPE from the A to the B interface is tagged with VLAN ID 35. The VLAN tag on traffic going in the opposite direction (from interface B to interface A) is dropped. Traffic between the Core Router and the Metro Router is encapsulated in an MPLS tunnel with label 250. All buffers are configured for 64 KBytes. Fig. 7 shows the traffic flows which are applied to the NSIM simulation topologies. The intention is to overrun the buffers at the core router ahead of the 5 Mbps traffic restriction on link3 (blue line). Over the course of the simulation time 5 thousand packets are generated. For the duration of the simulation, a constant traffic stream is generated by the trafgen traffic generator (Orange Line) attached to host1 and directed to the traffic sink attached to host 2. This traffic is the effective traffic being generated and received at the application layer.

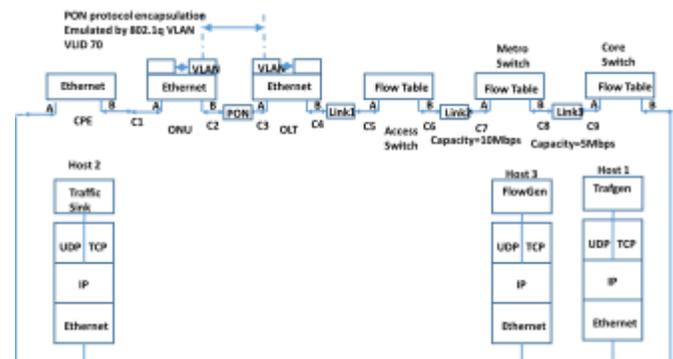


Fig. 8 FLATLAND architecture modelled in NSIM

For the simulations, this is set at 1.6Mbps. This is to be distinguished from the actual throughput at the data link layer which includes the application data encapsulated with TCP/IP layer headers and trailers. At $t=0.002$ seconds, the background traffic load (Grey Line) starts and ramps up quickly so as to swamp both or either of the limited bandwidth in the network topology or the buffers and queues in the network devices. The load traffic lasts until $t=1.000$ seconds, when it stops abruptly. Shown on the graph also is the restricted bandwidth capacity set at 5 Mbps. Fig. 8 shows the same scenario in Fig. 6 but configured for the FLATLAND architecture. All components such as datalinks and switches are configured with standard buffer sizes (64k), and link capacity and delays are the same as in Fig. 6, although many of the VLAN and MPLS encapsulation are eliminated, as they are not required in FLATLAND. We executed 7 scenarios which conduct a like-

for-like comparison between different configurations of buffer settings and queue sizes for the two architectures under inspection. Scenario A is the classic architecture with standard (64 KByte) buffer size, which can be directly compared to scenario D representing the FLATLANd architecture with standard (64 KByte) buffer size. Scenario B is the classic architecture with small (8 KByte) buffer size, which relates to similar FLATLANd scenario E. Scenario C is the classic architecture which uses the CoDel (Controlled Delay) queue discipline to disposes of packets older than 10ms, which can be compared to the CoDel FLATLANd architecture in scenario F. Finally, scenario G is the FLATLANd architecture with standard buffer and admission control applied at the traffic ingress. In the latter case, flows are not allowed on to the network if the traffic at the egress (cpe node) exceeds 3 Mbps (Green line in Fig. 7). Fig. 9 shows, for the target application traffic packets, the packet loss, jitter, delay and goodput (effective application throughput).

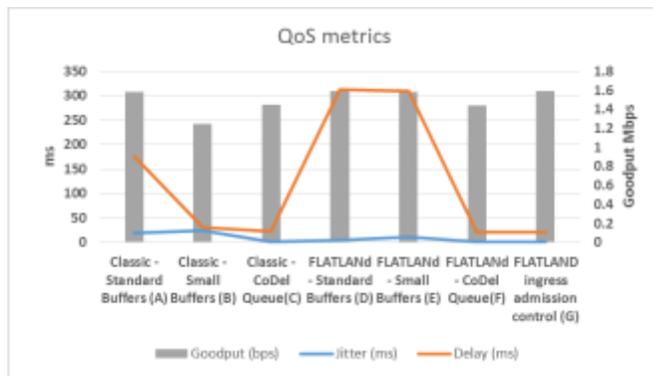


Fig. 9 QoS metrics

Fig. 10 shows lower level diagnostic data such as the number of packets affected by (1) rate limiting, (2) back pressure (3) that are dropped due to saturation of standard buffers and lastly (4) dropped due to aging.

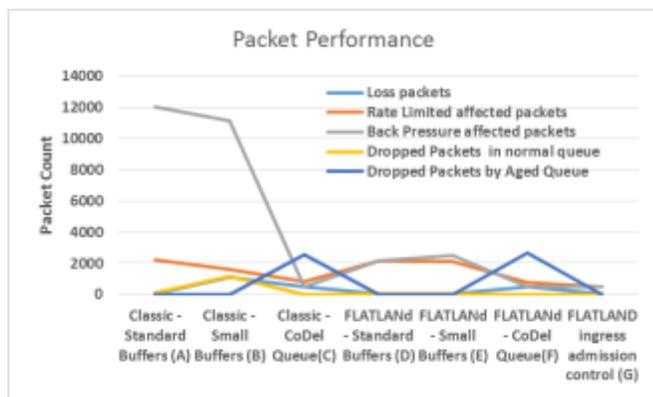


Fig. 10 Packet Performance

Back pressure is the term we use for the cascading upstream buffer overflow, when successive saturated upstream buffers don't drop packets and can't accept delivery of incoming packets. Overall, FLATLANd has good jitter, at the expense of end to end delay time (scenario D and E). We see the cause of the poor end to end delay down to rate limited events and back pressure events. Jitter deteriorates when the buffer size is

reduced. Jitter for the basic Classic architecture with either standard (scenario A) or small buffers (scenario B) is poorest. Again, this is due to severe rate limited events (12057 and 11134 packet events respectively) and back pressure events. The use of small buffers improves delay, however, this is at expense of goodput (lowest at 1.245 Mbps) and packet loss (21%). The introduction of CoDel queue aging (scenario C) at the Metro and Core routers in the standard buffers model improves jitter (1.719 ms) and delay (21.765 ms) significantly. Packets, that are buffered longer than 10ms are discarded. The use of the CoDel queuing discipline can have benefits also in the FLATLANd architecture (scenario F). There is a significant improvement in jitter and delay by comparison, however at the expense of goodput. The coincident use of CoDel queuing with FLATLANd is hypothetical, since FLATLANd is positioned as an alternative solution. This scenario is presented to demonstrate the incremental effect of introducing CoDel into the architecture. Of all the scenarios FLATLANd with admission control (scenario G) has the best packet loss (21), jitter (1.381 ms), goodput (1.593 mbps). NSIM shows, that on average, Ethernet encapsulation overhead accounts for 8.1% of traffic, IP overhead accounts of 11.5% of traffic and TCP or UDP overhead accounts for 4.7%. Each 802.1q or MPLS tunneling protocol adds between 2.3% and 3% overhead. Overall, the useful application payload occupies 70.8% of traffic in the classic architecture and 72.8% in FLATLANd. The overall amount of network operations, such as packet forwarding and routing, that is executed in the Classic Architecture is 40% higher than in FLATLANd. This is due to the additional VLAN tunneling across the access network and the MPLS tunneling across the metro-core network. The types of operations being executed also are more complex, with MPLS switching and routing, and VLAN pushing and popping being more complex to execute and more expensive in terms of calculation and processing. The classic architecture must execute 59'833 complex operations as opposed to 19'932 operations by FLATLANd over the course of the simulation. In classic architectures, the function of ARP address resolution is distributed to each Layer 2 broadcast domain, in particular at the terminating LAN and WiFi networks. The centralisation of ARP is already implemented in large datacenters. In the Portland model [38], it is assumed that each ARP requires 25 microseconds execution time with an ARP timeout of 60 seconds and each ARP packet is 28 bytes long. Using the model proposed, for a Flatland network operating over a region with 4 million terminating ONU's, each generating 1 Arp request per second, would require a 100 Core processors, which may be parallelized and distributed to 5 geographical areas in the network. In total, Arp queries and responses generates 896 Mbps of traffic. Because it is possible to transmit packets over a wide area, using the FLATLANd layer 2 addressing scheme, the source and destination hosts can identify each other using their Ethernet addresses. For the purposes of routing traffic over FLATLANd, the IP address of each host end becomes redundant. We can envisage a collapsed network protocol stack where the IP layer of the TCP/IP stack is removed, and the TCP or the UDP layer communicates directly with the Ethernet Layer. TCP and UDP continue to

provide the interface to the application layer for the purposes of end to end transport layer communication. We term these reduced protocol stacks as TCP over Ethernet (TCPoE) and UDP over Ethernet (UDPoE) respectively. Because the IP header in the packet is not used for routing by any device within FLATLAND, the removal of the IP packet encapsulation does not affect the functioning of the FLATLAND network architecture. There are advantages in removing a layer from the communications stack.

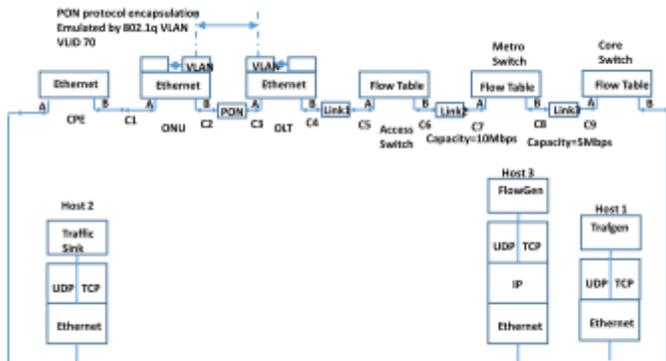


Fig. 11 FLATLAND Scenario D with UDPoE stacks

There is less packet processing required for encapsulation and de-encapsulation of packets. From our simulation measurements, the IP layer accounts for 11% of the total data exchanged at a line level. By removing the IP layer, Ethernet frames are shorter so less bandwidth is used to transfer data. In transferring the same application payload, switch and host buffers are less utilised, potentially leading to less network congestion. Because TCP over Ethernet (TCPoE) does not have a standard Ethertype, we create a new Ethertype (0x9999) and new binding between Ethernet and TCP in NSIM. Similarly, we create an Ethertype of 0x9998 for the UDPoE binding. Rerunning the scenario D tests in NSIM with the traffic source and sink hosts configured with collapsed UDPoE stacks (Fig. 11), we see that the improvement in protocol performance is significant.

```
###[ Ethernet ]###
dst  = 00:04:00:00:00:01
src  = 00:04:00:00:00:02
type = 0x8100
###[ 802.1Q ]###
prio = 0
id   = 0
vlan = 70
type = 0x9998
###[ UDP ]###
sport = 5344
dport = http
len   = None
chksum = None
###[ Raw ]###
load =
'ABCDEF GHIJKL MNOPQRST UVW'
```

Fig. 12 UDP over Ethernet (UDPoE) packet trace

With the removal of the IP protocol layer, useful payload has increased from 72% to 85% of total wire traffic. This results in a 25% improvement in delay and in 18% improvement in jitter when scenario 3 is re-run. Host3, which retains its full TCP/IP stack, still generates background traffic in the direction of Host2. This demonstrates that traditional TCP/IP traffic and the

envisaged TCPoE and UDPoE schemes can exist on the same network. Fig. 12 shows the Ethernet packet with a source MAC address of Host1 (00:04:00:00:00:01) and a destination MAC address of Host 2 (00:04:00:00:00:02), within which there is a 802.1Q VLAN tunnel (ID of 70). The ethertype is correctly interpreted as that of the new UDPoE encapsulation layer. Where usually there should be an internal IP layer, there now is a UDP datagram. From the perspective of service registration (Fig. 3), and not simulated here, all hosts require a Layer 2 binding phase. Only Host3 is necessitated to be authorised at Layer 3 through DHCP discovery and request. To verify that traffic is being relayed correctly without recourse to the IP network layer, we executed a packet capture at the PON network module in Fig. 11. We used the Scapy packet crafting tool to de-encapsulate the resulting pcap file.

VI. CONCLUSIONS

In this paper, we presented a flat layer 2 architecture for telecommunications networks that allows the removal of many components traditionally active in telecommunications architectures, while still retain much of the functionality for access and the delivery of service. Since a number of layers (such as PPOE tunneling) and component stacks (such as Broadband Access Services) are removed, there is less requirement for authentication and authorisation across junctions between these layers. With less layers (such as PPPoE tunneling) and component stacks (such as Broadband Access Services), the requirement for cross-layer authentication and authorization is greatly reduced. In addition, the FLATLAND architecture provides a separation between the provision of infrastructure, network services and Internet services by distinct entities, potentially enhancing efficiency of use of resources. Using Network simulation, we demonstrate that FLATLAND has good jitter and throughput compared to the classic architecture, but at the expense of end to end delay. In addition, FLATLAND presents lower packet loss and better throughput than the dominant (CoDel) remedial algorithm for Bufferbloat. While the topology we focused on, in the metro-access, was a tree structure, future work will focus on other topologies such as ring and hub which form the basis of Ethernet networks.

VII. ACKNOWLEDGMENT

This material is based upon works supported by SFI Grant No. 10/CE/I1853 and No. 14/IA/2527.

VIII. REFERENCES

1. B. Briscoe, "Flow rate fairness: Dismantling a religion," ACM SIGCOMM Computer Communication Review **37**, 63-74 (2007).
2. J. Warner, "Buffer Sizes of Common Routers and Switches. <https://people.ucsc.edu/~warner/buffer.html>," (2012).
3. J. F. Kurose, *Computer Networking: A Top-Down Approach Featuring the Internet, 3/E* (Pearson Education India, 2005).
4. G. Appenzeller, I. Keslassy, and N. McKeown, *Sizing router buffers* (ACM, 2004), Vol. 34.

5. T. Cloonan, "Minimising Bufferbloat and optimising Packet Stream Performance in Docsis 3.0 CMs and CMTSs," in http://snapon.lab.bufferbloat.net/~d/trimfat/Cloonan_Presentation.pdf, (2013).
6. V. Jacobson, "A rant on queues. A talk presented at MIT Lincoln Labs, Lexington, MA, 2006."
7. K. Nichols and V. Jacobson, "Controlling queue delay," *Communications of the ACM* **55**, 42-50 (2012).
8. J. Gettys, "Bufferbloat: Dark Buffers in the Internet," *IEEE Internet Computing* **15**, 96,95 (2011).
9. R. Braden, D. Clark, and S. Shenker, "RFC 1633. Integrated Services in the Internet Architecture: An Overview," IETF (1998).
10. R. Braden and L. Zhang, "Resource ReSerVation Protocol (RSVP)--Version 1 Message Processing Rules," 2070-1721 (1997).
11. J.-S. Li and C.-S. Mao, "Providing flow-based proportional differentiated services in class-based DiffServ routers," *IEE Proceedings-Communications* **151**, 82-88 (2004).
12. S. Blake, D. Black, M. Carlson, D. Davies, W. Wang, and W. Weiss, "RFC 2475. An Architecture for Differentiated Services," IETF (1998).
13. J. Roberts, "The clean-slate approach to future Internet design: a survey of research initiatives," *Annals of telecommunications* **64**, 271-276 (2009).
14. E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," 2070-1721 (2000).
15. M. Hassan and R. Jain, *High performance TCP/IP networking : concepts, issues, and solutions* (Pearson/Prentice Hall, 2004).
16. G. Armitage, "MPLS: the magic behind the myths [multiprotocol label switching]," *IEEE Communications Magazine* **38**, 124-131 (2000).
17. A. Chapman, "Automatic quality of service in IP networks," in *in Proceedings of the Canadian Conference on Broadband Research*, (Citeseer, 1997),
18. I. Stoica and H. Zhang, *Providing guaranteed services without per flow management* (ACM, 1999), Vol. 29.
19. J. Domzal, R. Wojcik, and A. Jajszczyk, *Guide to Flow-Aware Networking. Quality of Service Architectures and Techniques for Traffic Management* (Springer, 2015).
20. A. Kortebi, S. Oueslati, and J. Roberts, "Implicit service differentiation using deficit round robin," *ITC19* (2005).
21. A. Kortebi, S. Oueslati, and J. W. Roberts, "Cross-protect: implicit service differentiation and admission control," in *High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on*, (IEEE, 2004), 56-60.
22. J. Adams, "Flow State Aware signalling standardisation, and a proposal for alerting nodes or end-systems on data related to a flow," (2009).
23. J. Joung, J. Song, and S. Lee, "Flow-based QoS management architectures for the next generation network," *ETRI journal* **30**, 238-248 (2008).
24. R. Wójcik and A. Jajszczyk, "Flow oriented approaches to QoS assurance," *ACM Computing Surveys (CSUR)* **44**, 5 (2012).
25. B. Nandy, N. Seddigh, A. Chapman, and J. H. Salim, "A connectionless approach to providing QoS in IP networks," in *High Performance Networking*, (Springer, 1998), 363-379.
26. K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers," 2070-1721 (1998).
27. R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg, "PIE: A lightweight control scheme to address the bufferbloat problem," in *High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference on*, (IEEE, 2013), 148-155.
28. D. Hood, "TR-502 " (Open Network Foundation, 2014).
29. A. Aguado, V. López, J. Marhuenda, O. G. d. Dios, and J. P. Fernández-Palacios, "ABNO: a feasible SDN approach for multi-vendor IP and optical networks," in *OFC*, (2014),
30. W. John, A. Kern, M. Kind, P. Skoldstrom, D. Staessens, and H. Woesner, "SplitArchitecture: SDN for the carrier domain," *IEEE Communications Magazine* **52**, 146-152 (2014).
31. M. Tornatore, L. Gifre, B. Mukherjee, and L. Contreras, "abno-driven content distribution in the telecom cloud," (2015).
32. A. Napoli, A. D. #39, Errico, G. Ferraris, and M. Bohn, "Elastic optical networks: The vision of the ICT project IDEALIST," (2013).
33. "ONOS project," (2015).
34. A. Farrel, J. P. Vasseur, and J. Ash, "RFC 4655 - A Path Computation Element (PCE)-Based Architecture,," (2016).
35. P. Willis, "Network Functions Virtualization," in *SDN World Congress*, 2012),
36. R. Bifulco, "Rethinking Access Networks with High Performance Virtual Software BRASes,," in *EWSDN*, 2013),
37. F. Slyne and M. Ruffini, "FLATLAND: A Novel SDN-Based Telecoms Network Architecture Enabling NFV and Metro-Access Convergence. ," in *ONDM*, (2016).
38. R. N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric," in *SIGCOMM'09*, 2009),
39. S. Sharma, D. Staessens, and D. Colle, "Software defined networking: Meeting carrier grade requirements," (2011).
40. M. Ruffini, "Discuss: An End-to-End Solution for Ubiquitous Broadband Optical Access," *IEEE Communications Magazine* (2014).
41. ITU-T, "Rec. G.984. Gigabit-capable Passive Optical Networks (GPON)," (2003).
42. J. Matias, E. Jacob, N. Toledo, and J. Astorga, "Towards Neutrality in Access Networks: A NANDO Deployment with OpenFlow," (2011).
43. B. Heller, N. McKeown, and B. Lantz, "A network in a laptop: rapid prototyping for software-defined networks," (2010).
44. H. Shimonishi, "Virtualized network infrastructure using OpenFlow," in *IEEE.IFIP NOMS Workshop*, 2010),

Frank Slyne is completing his PhD at Trinity College Dublin in the research area of Metro Nodes as implemented in a flat-core and Long Reach Passive Optical Network (LR-PON) architecture. His interests are the application of software defined networks for the optimisation of capacity, energy and performance of telecommunications networks. He has worked for a number of years at Eircom where he was responsible for ISP platforms and systems. He has a B.E. (Elect) from UCC (Cork) and an M.Eng from DCU (Dublin).

Marco Ruffini received his M.Eng. in telecommunications in 2002 from Polytechnic University of Marche, Italy. After working as a research scientist for Philips in Germany, he joined Trinity College Dublin (TCD) in 2005, where he received his Ph.D. in 2007. Since 2010, he has been Assistant Professor (tenured 2014) at TCD. He is Principal Investigator at the CTVR/CONNECT Telecommunications Research Centre at TCD, currently involved in several Science Foundation Ireland (SFI) and H2020 projects, and leads the Optical Network Architecture group at Trinity College Dublin. He is author of more than 80 journal and conference publications and more than 10 patents. His research focuses on flexible and shared high-capacity fibre broadband architectures and protocols, network convergence and Software Defined Networks control planes.