EPJ Web of Conferences **295**, 07037 (2024) https://doi.org/10.1051/epjconf/202429507037
CHEP 2023

$P4_{flow}$: A software-defined networking approach with programmable switches for accounting and forwarding IPv6 packets with user-defined flow label tags

Carmen Misa Moreira[1], *Edoardo* Martelli[1], and *Tony* Cass[1]

[1]CERN - Conseil Européen pour la Recherche Nucléaire, Esplanade des Particules 1, 1211 Meyrin, Geneva, Switzerland, IT department CS group, email: firstname.lastname@cern.ch

Abstract. A comprehensive analysis of the HEP (High Energy Physics) experiment traffic across LHCONE (Large Hadron Collider Open Network Environment) and other networks, is essential for immediate network optimisation (for example by the NOTED project) and highly desirable for long-term network planning. Such an analysis requires two steps: tagging of network packets to indicate the type and owner of the traffic being carried and then processing of the tagged packets. The RNTWG (Research Network Technical Working Group) has defined a specification to identify the experiment and the application that originates a given network flow, named scitags (Scientific Network Tags) that is applied to the flow label field of the IPv6 header. We report here on the second step: processing of packets tagged according to this specification.

We developed $P4_{flow}$ as a software-defined networking approach by using P4 (Programming Protocol-Independent Packet Processors), a language for programming the data plane of network devices to accounting and process IPv6 packets with a scitags-based stamp in the flow label field, to understand the network utilisation and the applications used by the WLCG (Worldwide Large Hadron Collider Computing Grid) sites. With $P4_{flow}$, and exploiting the control plane capabilities provided by RARE/FreeRtr (an Open Source Network Operating System developed by the GÉANT community), we cannot only generate statistics concerning the traffic per experiment and application but can also, using an Intel Tofino P4-programmable ASIC Ethernet Switch, decide how to forward traffic matching defined flow labels. This latter capability is particularly interesting as we prepare for a future where LHC experiments will be sharing network links with other major science collaborations such as SKA and ITER.

1 Introduction

The scientific research community distributes the data generated by the Large Hadron Collider (LHC) experiments to sites participating in the Worldwide LHC Computing Grid (WLCG) for storage and analysis. To identify the various traffic streams of the multiple LHC experiments, generate statistics concerning network utilisation and optimise packet forwarding to WLCG sites, the Research Network Technical Working Group (RNTWG) has defined a packet labelling specification, known as scitags (for Scientific Network Tags) [1]. A detailed description of scitags is available in [2], a valuable contribution to this conference.

The scitags scheme sets out a standard for marking packets belonging to defined traffic streams from an individual experiment through the inclusion of a unique identifier within the 20-bit flow label field of the IPv6 header. These tags not only enable the collection of network utilisation data but also offer the potential to enable packet routing based on flows.

Table 1: Scitags packet marking specification.

Flow label field (20 bits)				
Bits 12-13 Entropy	Bits 14-22 (9 bits) Science Domain	Bit 23 Entropy	Bit 24-29 (6 bits) Application / Type of traffic	Bits 30-31 Entropy
-	DUNE: 8192 → 000100000	-	PerfSONAR: 2 → 000010	-
-	LHCb: 16384 → 001000000	-	Cache: 3 → 000011	-
-	ATLAS: 32768 → 010000000	-	Data Challenge: 4 → 000100	-
-	BelleII: 49152 → 011000000	-	Analysis Download: 9 → 001001	-
-	LSST: 73728 → 100100000	-	Data Access: 10 → 001010	-
-	ALICE: 81920 → 101000000	-	CLI Download: 13 → 001101	-
-	CMS: 98304 → 110000000	-	Data Consolidation: 15 → 001111	-
-	SKA: 114688 → 111000000	-	Production Download: 19 → 010011	-

At present, network devices cannot carry out accounting and forwarding of IPv6 packets based on the flow label field. However, the emergence of programmable switches and software-defined networks with data-plane programmability expands the capabilities of network devices. To leverage these capabilities, P4lang (Programming Protocol-independent Packet Processors) [3] is used to program the data plane of network devices, including those based on Intel's Tofino P4-programmable Ethernet Switch ASIC [4, 5].

P4lang offers a framework for defining how packets are processed in programmable switches, specifically at the data plane level, through the use of P4 programs. By describing the desired functionalities within a P4 program, a user can specify the header types of the packet, define the parser and de-parser functions for the ingress and egress pipelines and configure match-action tables to set packet attributes and facilitate traffic routing within the network [6]. Further, P4lang supports the use of intrinsic metadata and allows users to create their own custom metadata. P4 is a domain-specific language that can be used in a large variety of targets, including programmable network interface cards, FPGAs, software switches and hardware ASICs.

The Egdecore Wedge100BF-32QS [7, 8] is a P4-programmable switch based on the Tofino silicon programmable ASIC and designed for use as a high-performance data centre TOR (Top-of-Rack) or spine switch. With its bare-metal hardware architecture, it supports up to 6.4 Tbps of total bandwidth across a total of 32 ports, each capable of delivering 100 GbE connectivity through QSFP28 interfaces, and quad-pipe for programming the packet processing. Additionally, it incorporates an Intel x86 Xeon 2.0 GHz 8-core processor, 48 GB of RAM and a 2 TB SSD for storage.

For the control plane and the NOS (Network Operating System), we rely on FreeRtr [9], which is a key component of RARE, the Router for Academia, Research and Education [10, 11] project, funded by GEANT. FreeRtr is an open-source router operating system that manages the entries in the routing tables and also allows the forwarding of tables to DPDK or hardware switches via OpenFlow or P4lang.

The system described in this paper exploits the scitags packet marking specification to count and forward IPv6 packets based on the flow label field. As shown in table 1, the packet marking specification reserves 5 bits for entropy to match RFC 6436 [12], 9 bits are allocated to define the scientific domain and 6 bits are designated to define the application and

the type of traffic. For instance, traffic belonging to the ATLAS science domain from the perfSONAR (performance Service-Oriented Network monitoring ARchitecture) application is marked with the decimal number 32768 for the science domain and 2 for the application and type of traffic.

This paper is structured as follows. In section 1, we provide a summary of the motivation and objectives driving this workload. Section 2 outlines the network topology and scenarios employed to test flow label accounting and forwarding at both layer 2 and layer 3. The results obtained during the International Conference for High-Performance Computing, Networking, Storage, and Analysis (SC22) are presented in section 3, reflecting the comprehensive testing conducted throughout the conference week. In section 4,we discuss the utility and feasibility of traffic routing based on flow labels. Finally, section 5 summarises the conclusions drawn from our findings and outlines potential future directions.

2 Scenarios and methodology

In this section, we briefly describe the topology and the elements of the testbed network we built to test and demonstrate the feasibility of accounting and forwarding IPv6 packets based on the flow label field at both layer 2 and layer 3 by utilising FreeRtr as the control plane of an Edgecore Wedge100BFQS P4-programmable switch.

2.1 Access-list definition based on IPv6 flow labels

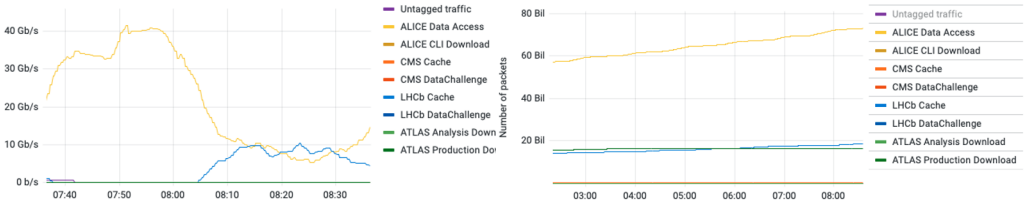
ACL sample 1:
Define an access-list that matches explicit flow labels.

```
access-list acl_flowlabels
 # Match on <Experiment> and <DataAccess>
 permit all any all any all flow 131076 & 163880   ▹ ATLAS
 permit all any all any all flow 65540  & 163880     ▹ CMS
 permit all any all any all flow 49152  & 163880 ▹ BelleII
 permit all any all any all flow 114688 & 163880     ▹ SKA
 # Match on <Experiment> and <perfSONAR>
 permit all any all any all flow 131076 & 261632   ▹ ATLAS
 permit all any all any all flow 65540  & 261632     ▹ CMS
 permit all any all any all flow 49152  & 261632 ▹ BelleII
 permit all any all any all flow 114688 & 261632     ▹ SKA
 # Permit the rest of the traffic
 permit all any all any all
exit
```

We show above how to define an acces-list on the control plane of a P4 programmable switch in order to distinguish traffic by experiment and traffic type. Each sequence within the access-list represents a permit statement that matches a specific flow label. To distinguish traffic by experiment and by science domain, bitmasks are used and the definition is by <flow label> & <bitmask> given in decimal notation. As shown in the example, the flow label 131076 is associated with the ATLAS experiment, with the bitmask 261632 is associated to the perfSONAR traffic type. To enable the accounting of IPv6 packets based on the flow label field, counters are declared in both hardware and software to match this access-list.

These counters are exported to a Prometheus database and displayed on a Grafana dashboard. Figure 1a shows the data rate in bits per second, while figure 1b shows the number of packets. We can observe that for the ALICE Data Access IPv6 flow label, data was transmitted at a

rate of 40 Gb/s, while for the LHCb Cache IPv6 flow label, the transmission rate reached 10 Gb/s. Similarly, the ALICE Data Access flow consisted of over 70 billion packets, and that for the LHCb Cache of around 20 billion packets.



(a) In bits/s (b) In number of packets.

Figure 1: Counters of an access-list.

2.2 Flow-label based accounting

Figure 2 shows the diagram of the network topology and the elements of the testbed built to test IPv6 flow label accounting at layer 2. The testbed is composed of a series of servers at a WLCG Site A site with BGP (Border Gateway Protocol) peering established with the Site B border router of both the LHCONE (Large Hadron Collider Open Network Environment) and LHCOPN (Large Hadron Collider Optical Private Network) networks.

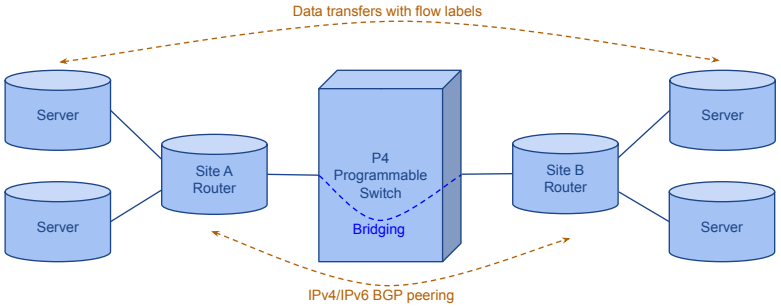


Figure 2: Testbed used for the IPv6 flow label accounting at Ethernet level (layer 2).

BRIDGING sample 2:
Define pure layer 2 bridging and ingress filtering.

```
interface sdn1.1000
    description [VLAN ID = 1000]
    bridge-group 1
    no shutdown
    no log-link-change
exit

interface sdn1.1001
    description [VLAN ID = 1001]
    bridge-group 2
    bridge-filter ipv6in acl_flowlabels          ▹ Apply ACL
    no shutdown
    no log-link-change
exit
```

Code sample 2 sets out how to configure the control plane of the P4 programmable switch to establish layer 2 bridges. Specifically, interface sdn1.1000, belonging to VLAN 1000, is configured as a pure layer 2 bridge. Similarly, interface sdn1.1001, belonging to VLAN 1001, is configured as a pure layer 2 bridge, but one accompanied by ingress access-list filtering.

With this testbed, we emulate a site-to-site network connection, where the P4 programmable switch is located transparently in the network without altering the content of the packets and without modifying the destination. Therefore, the traffic passes through the P4 programmable switch, enabling transparent and inherent packet counting.

2.3 Flow-label based routing

Having successfully shown that the use of scitags packet labelling enables the accounting of different traffic streams, we next investigated the possibility of using it to route traffic flows.

Figure 3 presents the network topology and the elements of the testbed built to test the IPv6 flow label forwarding at layer 3. The testbed consists of multiple servers at each end of the P4 switch to generate the traffic. For this test, DPDK (Data Plane Development Kit) was used to exploit the Network Interface Cards (NICs) to their maximum capacity to conduct high-performance tests and make use of all available resources for packet processing.

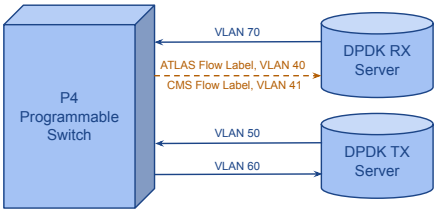


Figure 3: Testbed used for the IPv6 flow label forwarding at IP level (layer 3).

DPDK provides a collection of data plane libraries and polling-mode drivers for network interface controllers, enabling the offloading of TCP packet processing from the operating system kernel [13]. By offloading processing tasks, DPDK delivers higher computing efficiency and higher packet throughput compared to traditional interrupt-driven processing within the kernel. This framework creates a set of libraries for specific hardware/software environments through the creation of an EAL (Environment Abstraction Layer) [14] that hides the environment details and provides a standard programming interface for libraries, hardware accelerators and other operating systems.

PBR sample 3:
Define policy-based routing rules based on flow labels.

```
vrf definition ATLAS
exit

vrf definition CMS
exit

access-list acl_atlas
 | permit all any all any all flow 131076 & 163880    ▹ ATLAS
exit

access-list acl_cms
 | permit all any all any all flow 65536 & 261632      ▹ CMS
exit

ipv6 pbr CORE acl_cms   CMS    nexthop fd01:10::1
ipv6 pbr CORE acl_atlas ATLAS nexthop fd01:20::1
```

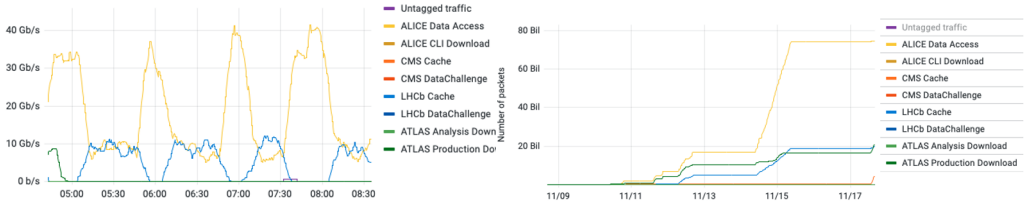
The example above describes the control plane configuration on the P4 programmable switch to define Policy-based Routing (PBR) rules, which are applied based on the conditions spec-

ified in the access-list. The access-list conditions establish whether a particular flow label value is associated with either the ATLAS or the CMS experiment, in accordance with the scitags packet marking scheme.

For each flow label, a VRF (Virtual Routing and Forwarding domain) and a VLAN (Virtual Local Area Network) are created where the PBR rules enable the forwarding of IPv6 packets based on their flow label. If a packet matches the ATLAS flow label, it will be forwarded according to the PBR rule configured for the ATLAS VRF. However, if the packet matches the CMS flow label, it will be forwarded based on a different rule defined for the CMS VRF, which specifies a different destination and route.

3 SC22 Demonstration and results

Packet marking on the DTNs is performed using either iPerf3 (the widely used tool for measuring TCP and UDP bandwidth performance), XrootD (an HEP-specific data transfer utility), both at speeds of up to 200 Gbps, or eBPF-TC [15] at a speed up to 100 Gbps depending on how the traffic is generated. Through these experiments, we successfully demonstrated the feasibility of using scitags-based packet and flow marking for both the accounting and routing of flows using the P4 programmable switch. The network statistics obtained from these tests are displayed on Grafana dashboards, providing a comprehensive overview of the network performance.



(a) iPerf3 and XRootD testing at SC22, swapping between protocols every 2h, in bits/s.

(b) iPerf3 and XRootD testing at SC22, swapping between protocols every 2h, in number of packets.

Figure 4: Counters of an access-list.

Figure 4 illustrates the results of demonstrations conducted during the International Conference for High-Performance Computing, Networking, Storage, and Analysis, also known as Super Computing, in November 2022 (SC22). In the first, traffic is measured in bits/s; the second shows the increase in the number of packets sent over time. The legends indicate the IPv6 flow labels for the different experiments and traffic types. During the tests, the DTNs generated traffic marked with different IPv6 flow labels, swapping between iperf3 and XRootD protocols every 2 hours.

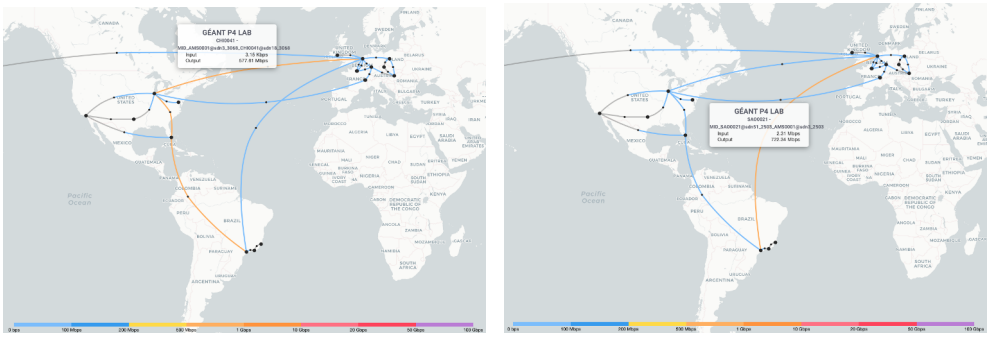
4 MultiONE

The utility of the LHCONE VRF setup by National Research and Education Network (NRENs) to support efficient data transfers between WLCG sites has been widely recognised and the LHCONE community has expanded to include other High Energy Physics (HEP) collaborations. LHCONE cannot, though, expand to cover all data-intensive science collaborations and so it is likely that other VRFs and ONEs will be established.

In such an environment, commonly termed "MultiONE", it would be beneficial for a site supporting multiple science domains to route traffic between the different "ONEs" (or VPNs,

Virtual Private Networks) based on the traffic type without needing to allocate distinct endpoints for the different domains.

To demonstrate the use of IPv6 flow labels to route and separate traffic across multiple ONEs we made use of GP4Lab [16], a worldwide P4 testbed developed as part of a GEANT-funded project, and the routing and forwarding procedures detailed in section 2.3. Three VRF instances were defined. The first serves as the default for IPv4 traffic and untagged IPv6 traffic, the second is dedicated to traffic belonging to the EXP-2 experiment, and the third is for WLCG traffic.



(a) Routing of EXP-2 traffic from São Paulo to Amsterdam via Chicago and Miami.

(b) Routing of WLCG traffic from São Paulo directly to Amsterdam.

Figure 5: MultiONE testbed in GP4Lab.

Figure 5 presents the results of MultiONE testing in GP4Lab, where it can be clearly seen that the traffic belonging to the flow label of the EXP-2 experiment is routed from São Paulo to Amsterdam via Chicago and Miami. On the other hand, traffic with the WLCG flow label is routed directly from São Paulo to Amsterdam. It should be noted that, although in our implementation all nodes have P4 programmable switches, only the site that generates the traffic actually requires a P4 programmable switch since it alone is responsible for routing traffic to the appropriate VPN based on the IPv6 flow label. This means that the burden of supporting a MultiONE environment falls solely on those sites that support multiple science domains; no special equipment will be needed at the majority of sites that support a single science domain or collaboration.

5 Conclusions

This paper presents a software-defined networking approach for accounting and forwarding of IPv6 packets with a scitags-based stamp in the flow label field. We tackled these goals by using P4 programmable switches as, for example, the Edgecore Wedge100BF-32QS and using RARE/FreeRtr for the control plane.

During the SC22 conference, we demonstrated that accounting and forwarding can be addressed at both layer 3 and layer 2 levels, using PBR rules and bridging along with an access-list to match specific flow labels. Furthermore, we demonstrated the feasibility, in a so-called MultiONE setup, to route traffic to different paths according to the tag found in the flow label.

Unfortunately, Intel has discontinued development of the Intel Tofino ASICs that were used in the current work. We are thus exploring alternative hardware solutions to enable us to continue our research.

References

[1] *Scientific network tags (scitags): initiative promoting identification of the science domains and their high-level activities at the network level*, https://www.scitags.org

[2] *Identifying and understanding scientific network flows*, https://indico.jlab.org/event/459/contributions/11321/

[3] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese et al., *P4: Programming protocol-independent packet processors* (2014), https://doi.org/10.1145/2656877.2656890

[4] *Intel tofino p4-programmable switch asic*, https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/tofino-series.html

[5] *$P4_{16}$ intel tofino native architecture - public version, application note* (April 2017), https://github.com/barefootnetworks/Open-Tofino/blob/master/PUBLIC_Tofino-Native-Arch.pdf

[6] *$P4_{16}$ language specification* (May 2017), https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html

[7] *Edgecore wedge100bf32qs quick start guide*, https://www.edge-core.com/_upload/images/Wedge100BF-32QS_QSG-R01_0706.pdf

[8] *Edgecore wedge100bf32qs datasheet*, https://www.epsglobal.com/Media-Library/EPSGlobal/Products/files/edgecore/Wedge_100BF-32QS.pdf?ext=.pdf

[9] C. Mate, *Freertr website*, http://www.freertr.org

[10] *Rare bitbucket*, https://bitbucket.software.geant.org/projects/RARE/repos/rare/browse

[11] *Rare/freertr documentation*, http://docs.freertr.org

[12] S. Jiang, S. Amante, B.E. Carpenter, *Rationale for Update to the IPv6 Flow Label Specification*, RFC 6436 (2011), https://www.rfc-editor.org/info/rfc6436

[13] *Dpdk programmer's guide, release 17.11.10* (February 2020), https://fast.dpdk.org/doc/pdf-guides-17.11/prog_guide-17.11.pdf

[14] H. Bi, Z.H. Wang, *Dpdk-based improvement of packet forwarding* (2016)

[15] T. Sullivan, *ebpf-tc github repository* (June 2022), https://github.com/hep-gc/Packet-Marking/

[16] *Gp4lab documentation*, https://wiki.geant.org/pages/viewpage.action?pageId=609058868