Towards a Privacy Preserving Data Flow Control via Packet Header Marking

Somnath Mazumdar
Department of Digitalization
Copenhagen Business School
Solbjerg Plads 3, 2000 Frederiksberg, Denmark
sma.digi@cbs.dk

Thomas Dreibholz
Centre for Resilient Networks and Applications
Simula Metropolitan
Pilestredet 52, 0167 Oslo, Norway
dreibh@simula.no

***Abstract*—Internet infrastructure is becoming ubiquitous thanks to the advancement in computing and the network domain. Reliable network communication is essential to offer good quality services, but it is not trivial. There are privacy concerns. Metadata may leak user information, even if traffic is encrypted. Some countries have data privacy preserving-related regulations, but end-users cannot control how their data packets should travel through networks. Even worse, the user cannot declare their privacy preferences. This paper presents an approach to tackle such privacy issues through data privacy-aware routing, where users can specify their preferences for packet routing using marking and filtering. Routing can work according to such specifications. It is implemented by P4, allowing a vendor-independent realisation with standard off-the-shelf hardware and open-source software components. We presented the initial experimental results of a proof-of-concept test on a unified cloud/fog research testbed.**

***Index Terms*—Cloud, Data, Fog, Packets, Privacy, Routing**

I. Introduction

Current global-level communication infrastructure encompasses intelligent heterogeneous hardware and a complex software ecosystem that helps Internet users to generate various data types in large quantities through their activities. Such data is stored and processed by a complex ecosystem of computer networks and distributed computing platforms. Users can see the travel path of its data packets using network path tracing tools, but they cannot control the route of the data packets. Such a complex scenario can reduce the fairness and transparency of how user data is stored, processed and accessed among the transit points. Improper management of online user data can lead to user-level privacy threats[1].

User data can be called "personal" if it can identify the user itself. It has been observed that websites can track their users and mislead them by providing deceiving information [1]. During such an online user tracking process, data is collected from users who visit web pages. Later, such information is linked to a user with a unique identifier. It has already been shown by [2] that web browsing histories can uniquely be linked to social media profiles using only public auxiliary information. That means adversaries can exploit such browsing histories. Due to employed data protection rules, user-generated data packets sometimes do not leave specific geographical regions. However, such data protection rules have impacted the web directly and indirectly. Privacy-related challenges also exist in Internet-of-Things (IoT) devices. Additionally, there are multiple unauthorised ways to access user data from the physical or network layer [3].

Informed users or domain experts want to know how data is transported, processed, and stored. Our proposed solution lets the informed user control its data packets. It is done by marking the packet headers and adjusting the data packet processing rules accordingly. It also tries to stop the exploitation of user data for monetary benefits by third parties without users' consent. In this paper, our research question is: *How can an informed user add its preference related to data packet movement to improve its data privacy?* To answer this, we have customised six bits of data packets via the packet tagging concept and implemented them using an abstract switch model. By letting users decide how packets should be flowing, the user can make informed decisions and can retain more control (to a certain extent) over their data. It may even unlock the full potential of programmable networks to improve user data privacy. In this paper our contributions are:

- We aim to improve the privacy[2] of online user data (related to data packet flows) by encoding custom mapping rules to process headers and fields of packets. We have done it using the Type of Service/Traffic Class field of the Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6).
- We have used programming protocol-independent packet processors (P4) [4] to process the packets as per the encoded instructions.
- Next, we have implemented the code targeting the Behavioral Model version 2 (BMv2) Simple Switch[3].
- Finally, we have reported the experimental results while testing in a cloud/fog research testbed.

II. A Primer on Packet Processing

A. Protocol-Independent Packet Processors (P4)

P4 is a domain-specific programming language for controlling the data plane of a programmable switch (refer to Fig-

[1]We define *privacy* as free from intrusion and having the ability to control one's data, while *security* refers to data protection against unauthorised access to user data. In some cases, privacy and security may overlap.

[2]"Improving privacy" means to stop/reduce unwanted access (such as snooping) of user data packets.

[3]BMv2: https://github.com/p4lang/behavioral-model.

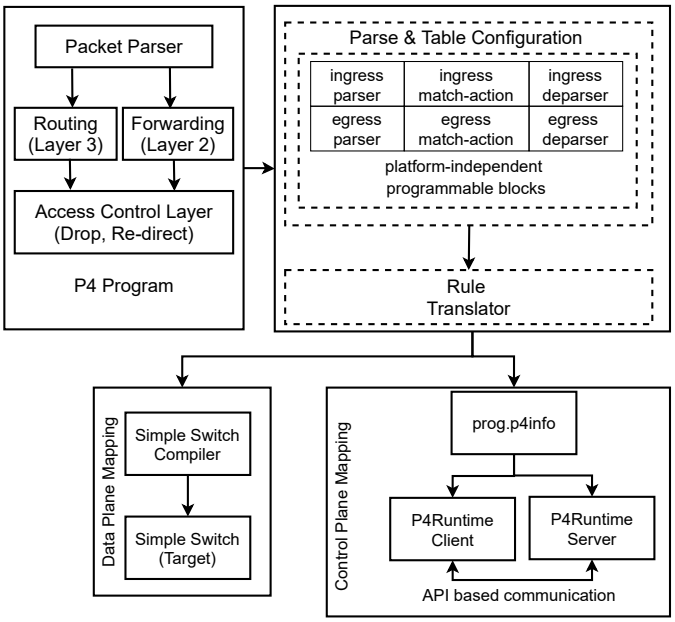

Figure 1. Block diagram representing P4 with Simple Switch as target.

ure 1). The switch can be either plain off-the-shelf hardware or a software implementation (such as Simple Switch in our case). Programming using P4 helps to encode customised rules into switching devices, allowing full flexibility for adapting the packet forwarding for certain setups and use cases. Particularly, P4 realises this flexibility in a platform- and vendor-independent way. P4 processes packets in multiple stages using five primary packet-processing abstractions. These are headers, parsers, tables, match-actions, and controls. Routing, forwarding, and access control layers have match-action tables based on which routing is done. During routing, if the routing of a packet is successful, it is moved from the L3 interface to the next level. Otherwise (during forwarding), it moves from the L2 switch to the next level. The access control layer decides whether to drop or redirect a packet. Apart from these abstractions, P4 also offers six programmable, platform-independent blocks. They are ingress parser, ingress match-action, ingress deparser, egress parser, egress match-action, and egress deparser (refer to right-top of Figure 1). As per the programmer's specification, the ingress parser extracts packets into headers. Next, the ingress match-action decides how packets will be processed and queued for egress processing (ingress deparser). After dequeuing, packets are processed by egress match-action. Finally, packets are deparsed from headers into a bit of representation following the egress deparser specification. The P4 configuration consists of match-action tables of ingress and egress ports. Then, the switch handles a packet according to a matching entry in the match-action table. For instance, a packet can be altered based on *i)* source and destination addresses, *ii)* protocols (such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Stream Control Transmission Protocol (SCTP), Datagram Congestion Control Protocol (DCCP)) and *iii)* port numbers. This can be done by rewriting the information in the *DiffServ Code Point* (DSCP) bits of the Type-of-

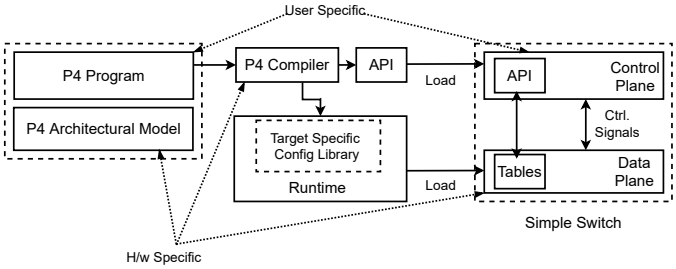

Figure 2. An illustration of Simple Switch implementation.

Service (TOS)/Traffic Class fields of the IPv4/IPv6 headers (or additional headers). Furthermore, it can be used to tag packets based on application, content, or privacy requirements to process the packets differently. The control plane is the software part that can be used to adjust the switch's behaviour. In particular, the data plane can forward a complete or parts of a packet to the control plane for custom handling (such as to learn media access control (MAC)-to-port mapping for a newly seen device) or further analysis of a certain packet. Also, the control plane can modify the lookup tables used by the data plane. For instance, to let the data plane (fast, in hardware) use a table to filter or mark certain packets without delegating action to the control plane (slow, in software).

B. Simple Switch

Simple Switch can be used to implement a custom P4-programmable architecture. It helps to customise the switching behaviour by separating the control and data plane (refer to Figure 2). The implementation is based on the standard BMv2 library. In the Simple Switch model, a queue exists between ingress and egress segments. Packets are parsed based on the specified protocol headers in the parser. While deciding on the destination, deletion of the packets and modification to the packet (i.e. headers and their fields) are done at ingress and egress segments. In Simple Switch, we can set different parameters (such as the length of the queue and queue throughput). Operational data can be collected from the counters and registers implemented in the Simple Switch. P4 code based on Simple Switch is target-independent. The P4 compiler processes a P4 program and an architectural model definition. It generates the input for the control and data plane of the target, which is a Simple Switch in our case. It separates components between user-specific and hardware-specific. The target realises appropriate mechanisms for the interaction between the control and data plane. Such mechanisms include table modification by the control plane or data forwarding from the data plane to the control plane.

III. Proposed System and its Implementation

Here, we discussed which field we have extended in Subsection III-A. Subsection III-B discusses how our proposed packet tagging concept works. In Subsection III-C, we have mentioned how the standard Simple Switch is extended using P4.

Table I
EXAMPLE OF USING TOS/TRAFFIC CLASS FOR PRIVACY SETTINGS.

Bit	Description
7	Allow EU/EEA area (GDPR)
6	Allow US-controlled area (Americas, UK, AU, NZ, UA, etc.)
5	Allow China-allied area (China + New Silk Road countries)
4	Allow Russia-allied area
3	Allow Iran-allied area (North Africa, Middle East)
2	Allow Saudi-Arabia-allied area (North Africa, Middle East)

A. Type-of-Service (TOS) and Traffic Class

The Type-of-Service (TOS) is a field of the IPv4 header, while the Traffic Class is a field of the IPv6 header. Their contents are the same despite their different names. The 8-bit field consists of a 6-bit DSCP (specifically from Bit 2 to Bit 7), and two bits are used for Explicit Congestion Notification (ECN [5], Bit 0 and Bit 1). These two ECN bits are used by transport protocols (such as TCP, SCTP, and DCCP) and routers for signalling congestion conditions. However, the 6-bits of the DSCP can be freely used to mark packets for different parameters, such as Quality-of-Service (QoS) management, by assigning priorities or service classes. The interpretation of these bits may be according to the customs rules of network administrators or ISPs due to the non-standardisation.

B. Tagging of Packets

There are two approaches for tagging packets with privacy information. They are using the DSCP (simpler approach) and adding additional headers (complex approach).

1) DSCP-Based Tagging: Using the DSCP, it is possible to write information into the IPv4/IPv6 header directly. It can be done as a part of the TOS/Traffic Class field without increasing the size of the packets. There will not be any problems with the Maximum Transmission Unit (MTU) of the underlying networks as the packet size remains unchanged. However, only 6-bits in the DSCP can be freely allocated. It is possible to realise a basic specification of privacy by region using the DSCP while, on the other hand, not increasing packet size and avoiding MTU issues. Our proposed mapping is presented in Table I. An actual commercial implementation may adapt to the global/political/financial situation. A few of them might be *i)* the mapping of areas may change over time (e.g. integration of Donetsk into Russia); *ii)* areas may overlap (e.g. countries allied to both Russia and China)[4]. Our basic idea for the DSCP is as follows:

- Default DSCP is 000000_2.
- Local country is always allowed (no special settings are needed).
- Allowance for up to six economic/political areas can be given (see Table I) by setting the corresponding DSCP bit. The default DSCP 000000_2 disables all (except the local country), while 111111_2 allows all.

[4]Refer to disclaimer in Section VI.

The sender (or router) can mark their outgoing packets with specific DSCP values for the desired privacy requirements, allowing or prohibiting certain regions.

2) Additional IP-Header-based Tagging: [6] specifies a "Security Option" for IPv4 to provide confidential information for a packet to be used in routing. The packet content remained unchanged (i.e. not encrypted/signed on Network Layer). This option would just have marked the "interesting" packets for an observer instead of providing the intended confidentiality. However, we can reuse the basic idea by adding an expression of privacy requirements into packets. It would be possible to make more detailed privacy specifications with more space. There is no freely usable space left in the IPv4/IPv6 header, except for the 6 bits of the DSCP. Thus, additional space is needed. The header length of IPv4 is variable and allows further options to be specified as part of the IPv4 header. For IPv6, an extension header with options has to be added. In both cases, the overhead and the length of the packet increase. In the case of an original packet of MTU size, the sending transport protocol has to reduce the maximum segment size (MSS) to ensure that the modified packet can fit into the MTU to avoid fragmentation of the IP layer. Privacy specifications are a list of items. Each privacy specification item may contain the following:

- `Action`: It can have binary values (Allow means 1 and Deny means 0). The Action (Allow/Deny) specifies whether the Entity Type (such as a country) is allowed or prohibited.
- `Match`: Match can either be Specific (represented by 0) or ALL (denoted by 1). An exact value for the Entity Type (e.g. a specific country) will be given when Match is set to Specific. Otherwise, for Match being ALL, any value will match (e.g. any country). Furthermore, ALL can be used to specify default (e.g. Deny ALL countries).
- `Entity Type`: It can have 64 values (from 0 to 63). We have considered four types of entities. They are Region, Country, Autonomous System (AS) and Private Enterprise Number (PEN) (relevant to our platform, see Figure 3 for details). Further Entity Types could be added when necessary.
  - `Region (0x00)`: Value is 1-byte with an economical/political region ID (similar to Table I), here allowing up to eight regions (with 8 bits).
  - `Country (0x01)`: Value is 2-bytes with ITU-T E.212 Mobile Country Code (MCC[5]). For countries with multiple MCCs (e.g. United States), the lowest value must be used.
  - `Autonomous System (AS) (0x02)`; Value is 4-bytes with IANA AS number[6], corresponding to the network. It also covers ISPs.
  - `Private Enterprise Number (PEN) (0x03)`: Value is 3-bytes with IANA PEN[7].

[5]MCC: https://en.wikipedia.org/wiki/Mobile_country_code.
[6]AS: https://www.cidr-report.org/as2.0/autnums.html.
[7]PEN: https://www.iana.org/assignments/enterprise-numbers/enterprise-numbers.

- `Entity Value`: (0...n bytes) according to the Entity Type.

3) Working Explanation: The list of specifications is processed sequentially when a packet is handled according to the markings. It is worth noting that marking by P4 is independent of marking privacy. For each `Entity Type`, the *first* matching rule specifies the processing of the given `Action`. For instance, there could be rules matching some specific allowed countries, with an additional rule having `Match` set to *ALL* for denying any other country. The packet may not be forwarded as soon as a rule leads to *Deny*. A packet is only forwarded if no matching rule leads to Deny (refer to Table V). Below, we are presenting three fictitious use cases:

- **Use Case A:** Allow only China and Russia, without Cisco equipment:
  - `Allow` "Country: 460 (China)",
  - `Allow` "Country: 250 (Russia)",
  - `Deny` "Country: ALL",
  - `Deny` "PEN: 9 (Cisco)".
- **Use Case B:** Allow only North America, without Huawei equipment and not via China Telecom:
  - `Allow` "Country: 310 (USA)",
  - `Allow` "Country: 302 (Canada)",
  - `Deny` "Country: ALL",
  - `Deny` "PEN: 2011 (Huawei)",
  - `Deny` "AS: 4809 (China Telecom)".
- **Use Case C:** Allow all countries except Sweden:
  - `Deny` "Country: 240 (Sweden)".

C. Implementation in Simple Switch using P4

Both approaches mentioned in Subsection III-B can be implemented with P4. To keep the packet sizes unchanged, we have implemented the DSCP approach. Adding custom headers in P4 is relatively easy, but the increased packet length would affect the performance of upper-layer protocols. It means transport protocols (such as TCP) must reduce their MSS due to the increased overhead. Using the DSCP approach avoids this by keeping the packet lengths unchanged. A firewall could also realise a pure DSCP approach. However, compared to P4, this would *i)* remove the flexibility for more advanced packet processing, and *ii)* make the approach dependent (such as vendor lock-in) on a certain firewall system (e.g. NETFILTER for Linux, PACKET FILTER under FreeBSD, Cisco IOS). Custom deep packet inspection is a motivation for using P4. It can apply customised rules to handle marking, perform filtering, and create alerts. Such customised rules can be provided as "modules" and deployed by the controller plane. With P4, this is made in an open, standardised and vendor-independent way on off-the-shelf devices.

In our target switch, a P4 program realises the switch functionality based on the Simple Switch. The switch provides basic functionality, e.g. handling a MAC-to-port forwarding table. It also uses the Privacy Marking Table (shown in Table II). The user configures this table according to their privacy preferences. Based on *Egress port* (towards the Internet; here: `port 0`) and *VLAN ID*, it defines the DSCP to be set. The

Table II
PRIVACY MARKING TABLE.

Egress Port	VLAN ID	Traffic Class Marking
0	4000	0xFC (Allow all)
0	4001	0x00 (Allow none, except local country)
0	4002	0x80 (Allow EU/EEA)
0	4003	0x30 (Allow China+Russia-allied area)

Table III
PRIVACY ENFORCEMENT TABLE.

Ingress Port	VLAN ID	Prohibited Marking
0	4003	0x04 / 0x08 / 0x0c / 0x40 / 0x44 / 0x48 / 0x4c / 0x80 / 0x84 / 0x88 / 0xc0 / 0xc4 / 0xc8 / 0xcc (drop if not marked as China-allied and/or Russia-allied allowed)

security concept of our local cloud setup is to set up different virtual LANs (VLAN) for various applications, providing a strict separation between unrelated systems. For instance, an application requiring remote cloud/fog connectivity should not be in the same network as an unrelated application requiring no remote cloud/fog connectivity or connectivity to a different region. Therefore, the privacy issue of an application in one VLAN does not affect the privacy of unrelated ones. We apply the region marking per VLAN ID as proposed in Table I. Note that we specify DSCP as part of the ToS/Traffic Class. It is shifted to the left by two bits. The lower two bits used by ECN (see Subsection III-A) remain unchanged, so as to not interfere with ECN marking, i.e.:

$$\text{TrafficClass}_{\text{new}} := (\text{TrafficClass}_{\text{old}} \ \&\ 00000011_2) \mid \text{Marking}.$$

A Privacy Enforcement Table (as shown in Table III) can be used on P4 switches/routers within networks, including cloud/fog sites. It can be used on a router to the Internet to read the markings from the packets and handle them according to their marking. For example, an ISP peering with a Chinese-Russian cloud provider would forward packets marked with allowance for China-allies and/or Russia-allies. Otherwise, the packet will be dropped.

D. Is ISP Support Necessary?

An ISP may change the DSCP for its purposes. It is good to assume that ISPs may not support privacy-preserving rule(s). Therefore, a simple solution is to tunnel many privacy-preserving (tagged) connections between ISPs via a virtual private network (VPN) tunnel. By aggregating multiple flows into a single VPN tunnel, the activities of single users cannot be derived (e.g., by packet size, packet inter-arrival times, protocol, and metadata). An unwanted entity might sniff even encrypted VPN traffic. In the worst case, our proposed privacy-preserving functionality is only provided by the user's switch and the remote cloud/fog switch. Then, adding dummy traffic and padding packets to MTU size can be used to mask the actual user's traffic behaviour to ensure privacy. Our solution with VPN tunnels enhances privacy even without direct support from end-user ISPs and intermediate network

Table IV
CLOUD/FOG RESEARCH TEST SITES WITH LOCATION, ISP NAMES.

Abbreviation: Site, Location	ISP 1	ISP 2
SRL: Simula Research Lab., Oslo, NO	Uninett	–
NTNU: NTNU Trondheim, Trondheim, NO	Uninett	PowerTech
UiA: Universitetet i Agder, Kristiansand, NO	Uninett	PowerTech
UiB: Universitetet i Bergen, Bergen, NO	Uninett	BKK
UiO: Universitetet i Oslo, Oslo, NO	Uninett	–
HU: Hainan University, Haikou, CN	CERNET	Unicom
KAU: Karlstads Universitet, Karlstad, SE	SUNET	–
UDE: Universität Duisburg-Essen, Essen, DE	DFN	–

[1] BKK, CERNET, DFN, SUNET, Uninett offers 1000000 Kbit/s for both, download and upload.
[2] PowerTech supports 6000 Kbit/s as download and 512 Kbit/s as upload.
[3] China Unicom offers 50000 Kbit/s for both download and upload.

Table V
CONNECTIVITY VALIDATION

From VLAN	To			
	NO (local)	EU/EEA	China+Russia	Others
4000	Yes	Yes	Yes	Yes
4001	Yes	No	No	No
4002	Yes	Yes	No	No
4003	Yes	No	Yes	No

providers. In a simple case, there would be a VPN endpoint on the user's premises and the networks of the used cloud providers. In addition to VPN tunnels, a rule-based framework can ensure network providers' adoption. However, this may be challenging to achieve depending on the involved entities. So, a VPN-based solution is an easy-to-achieve deployment possibility. In the setup, the router to the Internet can provide the VPN endpoints and route packets to either the Internet directly or via its configured VPNs, according to the packet markings. Routing depending on DSCP and VPN configuration are standard features of routers. There exist multiple VPN solutions, but it adds overhead similar to adding additional headers.

For security reasons (strict or loose), source routing is disabled. Any valid source routing concerning privacy would require topology information of remote networks. Such knowledge means information about the region, country, and privacy rules. It is only sometimes available. An ISP only knows *its own* ASs. Multi-Protocol Label Switching (MPLS), in the Data Link Layer below IP, is generally unavailable to end-users. However, ISPs can use MPLS paths as virtual leased lines to forward traffic on specified paths. In general, VPNs are the universal solution.

IV. EXPERIMENT AND RESULTS

A. Experimental Setup

A cloud/fog research testbed has been used for our experiment, which is part of the NORNET CORE infrastructure [7]. Our local private cloud setup consists of devices connected to a P4-based customised switch. `Port-0` of this switch is connected to a router, connecting the setup over the Internet to three remote fog sites (each in Germany, Sweden and China) and one local (in Norway). For more details on each connected ISP's maximum download/upload network speed, refer to Table IV.

All remote cloud/fog VMs, routers and devices, including the P4 switch, run on UBUNTU LINUX. The local private cloud setup ("Home") consists of devices, a P4 switch and a router, all of which are running on a local Dell server[8]. Our cloud platform is built on Kernel-based virtual machines. All cloud servers have at least one of 4-core Intel Xeon E5606 CPU at 2.13 GHz. These test sites have IPv4 and IPv6 connectivity. We provide details about the routing between the sites in Subsection IV-B and the round-trip times (RTTs) in Subsection IV-D. Our routing and RTT measurements are performed using the HIPERCONTRACER [8] tool, which runs the ICMP/ICMPv6 Echo Request/Reply measurement series to record routes (Traceroute) and RTTs (Ping). Bandwidth measurements are performed using the NETPERFMETER [9] tool.

B. Routing between Test Sites

We conducted routing and RTT measurements between the local private cloud setup (at SRL) and the other sites in both directions for one month. The measurements are performed approximately every ten minutes via IPv4 and IPv6 over all ISP combinations. Figure 3 shows the observed links and their mapping to ASs and geo-locations. Many different routes are observable, despite having a relatively simple setup of sites primarily in Norway and additional sites in Germany, Sweden and China. That is, changes in the routing over time are common. From the perspective of privacy, it is also crucial to note that connectivity involves third-party countries and geographic areas [10]. Many observed routes involve routing between Europe and China via the United Kingdom and the United States, instead of taking the direct route via Russia. While this routing is likely due to business contracts between ISPs, there may be better choices for some users concerning privacy. Such findings *motivate* work on giving users the possibility to influence the choice of underlying networks used in their communications.

C. Connectivity Validation

To validate our proposed solution, we checked that the markings of our Privacy Marking Table (see Table II), and the rules by the corresponding Privacy Enforcement Tables on our Internet router (see Table III as example for VLAN 4003), are handled correctly. Table V shows the resulting connectivity. It says, if a destination is allowed, packets are forwarded. If a relation is prohibited, e.g. packets to Germany (EU/EEA, but not Russia/China-allied) from VLAN 4003, the corresponding packets get dropped (resulting in 100% packet loss for the corresponding flow). It proves that our system is working as intended.

D. Round Trip Time (RTT) Results

Next, we checked the impact of the P4-switch on the RTT between a device in the home network and cloud servers on

[8] with Intel Xeon E5-1650 CPU at 3.20 GHz having 6 dual-threaded cores.

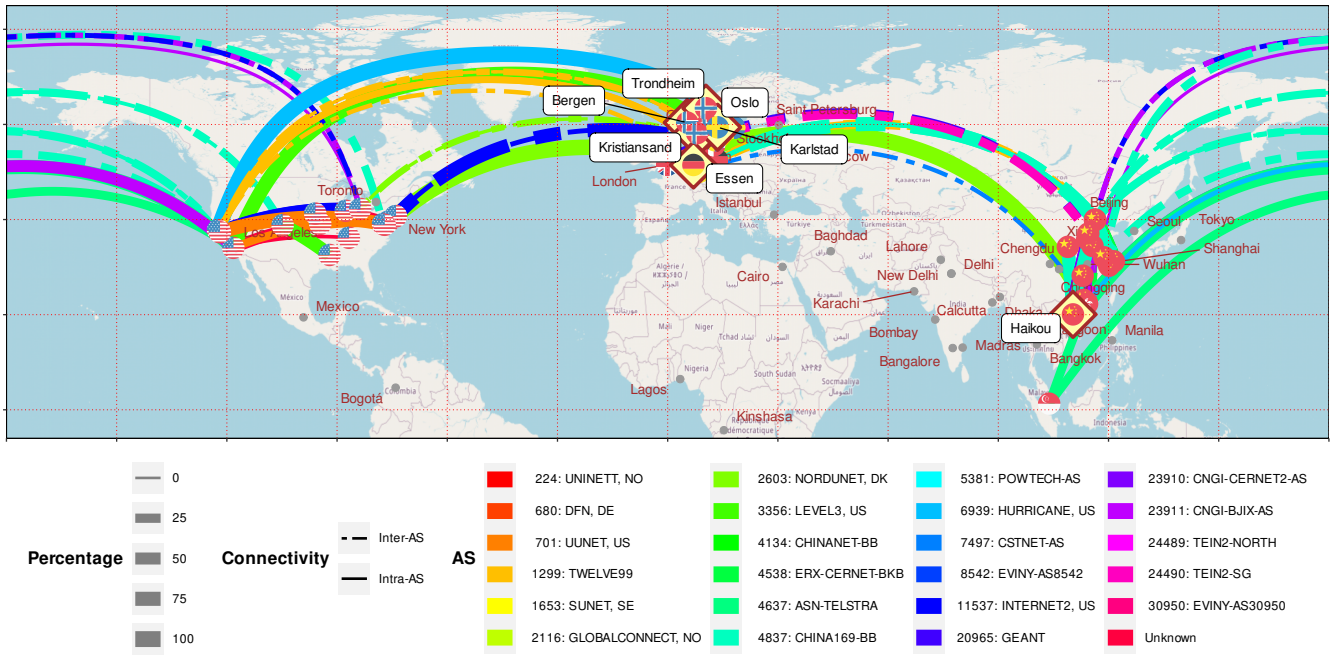

Figure 3. Observed AS between the sites of Table IV.

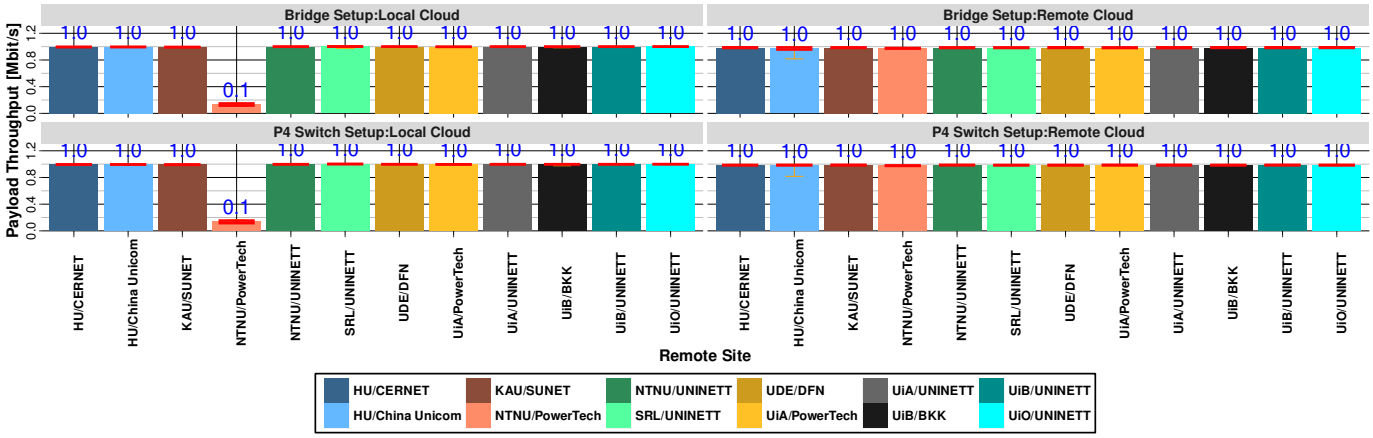

Figure 4. Bidirectional Scenario: 1 Mbit/s Symmetric UDP Flow.

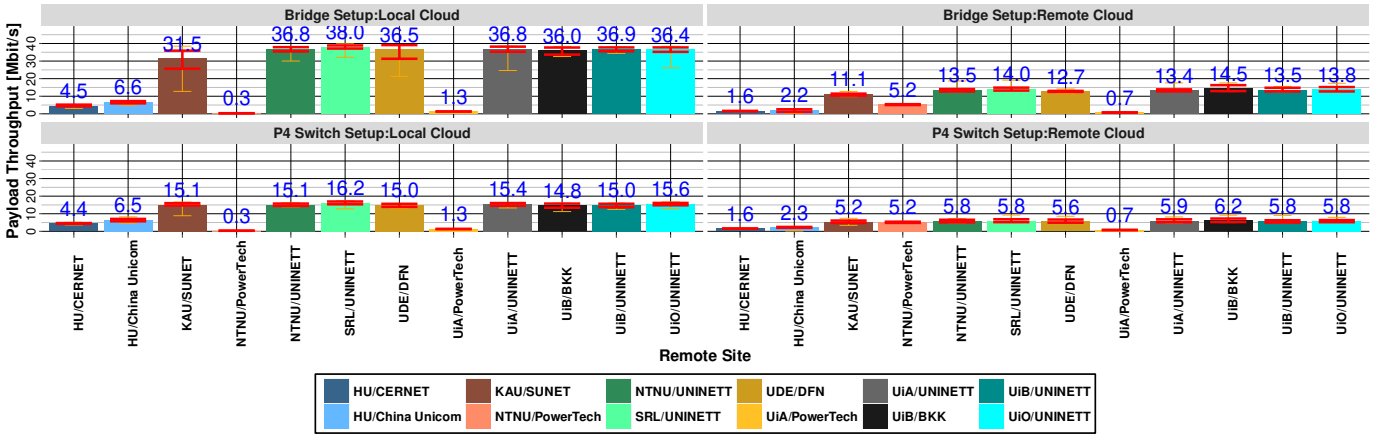

Figure 5. Bidirectional Scenario: Saturated TCP Flow.

Table VI
RTT MEASUREMENT RESULTS: LINUX BRIDGE VS. P4 SWITCH CONSIDERING BOTH IPV4 AND IPV6 WHILE REPRESENTING RTT OF 900 SAMPLES.

Destination	Plain Linux Bridge						P4 Switch with Marking and Filtering					
	IPv4			IPv6			IPv4			IPv6		
	Mean	$Q_{10\%}$	$Q_{90\%}$	Mean	$Q_{10\%}$	$Q_{90\%}$	Mean	$Q_{10\%}$	$Q_{90\%}$	Mean	$Q_{10\%}$	$Q_{90\%}$
HU/CERNET	341.2	340.4	341.9	341.2	340.4	341.9	352.4	347.4	356.2	349.5	345.2	352.7
HU/China Unicom	202.2	190.6	231.9	272.1	258.2	300.0	205.7	198.1	208.5	269.8	262.9	272.5
KAU/SUNET	32.5	31.7	33.2	32.8	32.0	33.5	42.9	37.9	46.3	39.7	35.7	42.8
NTNU/PowerTech	24.2	21.1	24.3	24.5	21.4	24.1	32.4	26.0	35.9	29.7	24.6	32.8
NTNU/UNINETT	14.7	13.9	15.3	14.7	13.9	15.3	20.2	16.5	22.5	18.7	16.0	21.0
SRL/UNINETT	5.2	4.6	5.8	4.7	4.0	5.4	13.3	8.3	16.8	10.3	5.8	13.7
UDE/DFN	32.2	31.4	32.9	37.9	37.1	38.5	42.9	37.7	46.2	45.4	41.3	48.6
UiA/PowerTech	24.2	22.9	24.7	–	–	–	31.7	27.3	34.8	–	–	–
UiA/UNINETT	12.2	11.5	12.7	12.3	11.5	12.8	17.2	14.0	19.8	15.8	13.3	17.7
UiB/BKK	12.4	11.6	13.2	12.5	11.7	13.2	19.5	15.2	22.2	17.6	14.2	20.3
UiB/UNINETT	12.3	11.6	12.9	12.5	11.8	13.1	16.8	13.7	19.5	15.6	13.2	17.5
UiO/Broadnet	18.6	14.9	19.6	19.3	15.0	19.9	27.2	21.3	30.3	25.1	19.3	28.2
UiO/UNINETT	7.0	6.1	7.4	7.0	6.2	7.4	13.9	9.3	17.3	11.2	7.5	14.3

the Internet, covering Norway (EEA), Germany and Sweden, and China. Per server, around 900 samples have been recorded using HIPERCONTRACER. We computed the mean, and 10%- and 90%-quantiles. Table VI presents the results for using a standard Linux bridge (without marking or filtering) in comparison to using our P4-switch. Since the results for the VLANs do not vary significantly (except for the intentionally blocked connections already examined in Subsection IV-C), the presented results only show VLAN 4000 (where all connections are allowed). The P4-switch, sets the *Allow All* marking of `0xFC` in the Type-of-Service/Traffic Class fields of the packets. Only the DSCP is set according to Table II, and the ECN bits remain unchanged. The data plane has to compute and verify the IPv4 header checksum and set the DSCP bits of the Type-of-Service/Traffic Class field according to the Privacy Marking Table. Finally, the new IPv4 header checksum is computed. IPv6 has no header checksum, so it is useful to distinguish between the two protocols here.

As shown in the results, adding the P4 Simple Switch adds around 6 ms to 12 ms of additional RTT to the IPv4 communications. Due to the checksum computation, the added cost is a bit more for IPv4 compared to IPv6. For IPv6, the difference is a bit smaller, with around 5 ms to 10 ms. Such additional RTT is insignificant for most latency-tolerant applications. It is unlikely that a highly RTT-critical application would use remote cloud/fog resources since the transport on the Internet between countries takes significantly more time (e.g. in Table VI: RTT for IPv4 of around 32 ms for UDE/DFN, Germany; or even around 202/341 ms for HU, China, depending on the used ISP). On the other hand, P4 adds total flexibility to packet handling by custom P4 programming.

E. Throughput Measurement

For UDP, we used a bidirectional flow with 25 frames/s at 5000 B of payload in packets of 1500 B (Ethernet MTU) (i.e. a symmetric payload bandwidth of 1 Mbit/s) over IPv4. Figure 4 shows the *received* application payload throughput for using a Linux bridge (upper part) *without* any marking or filtering compared to the P4-switch with privacy marking and DSCP filtering on the router (lower part). The local cloud-side view (local-site reception throughput) at SRL is shown on the left-hand side, while the remote cloud-side view (remote-site reception throughput) is shown on the right-hand side. Each measurement takes 30 seconds, and the bars show the average over 30 runs. The red error bars present the 10%/90%-quantiles, while the thin error bars show the absolute minimum and maximum of the runs. First, we can see that our system is working as required. From the reception throughput perspective, regardless of direction (to local/remote site), the P4 switch results are similar to the bridge results. In this setup, we use a non-saturated UDP flow and have not reached the P4-switch's processing capacity limit. Instead, the network is a bottleneck. It can be particularly observed in the flow from NTNU/PowerTech to SRL. Here, the received application payload throughput at the local cloud site at SRL (see the left-hand side of Figure 4) is only 0.1 Mbit/s. Frames of 5000 B need to be segmented (by the tool) into four UDP packets (due to MTU of 1500 B). Even if only one of the four packets gets lost, the whole payload of 5000 B is lost. The bottleneck is the upload at NTNU/PowerTech which is a 6000/512 Kbit/s Asymmetric Digital Subscriber Line (ADSL) subscription with only 512 Kbit/s of upload. The reverse direction (see the right-hand side of Figure 4) has sufficient bandwidth for the 1 Mbit/s flow.

We use a saturated, bidirectional TCP flow for the next measurement (shown in Figure 5) to show the limits of our P4 setup based on Simple Switch. We can see that the ADSL connections (such as NTNU/PowerTech) are bottlenecks for both the bridge and the P4 switch setup. The results for both setups are then mostly similar. However, for faster connections, the P4 switch setup introduces packet losses, negatively impacting TCP performance. TCP assumes the losses as an indication of network congestion, leading to a reduction of the flow bandwidth as P4 Simple Switch reaches its limit (by fully utilising one of the 6 virtual CPU cores of its VM). It

can be seen that flexibility comes with a price.

One P4-switch can be used for each household with a preferred configuration which does the marking using six free DSCP bits. We avoid adding additional headers that reduce the MSS and add more overhead while implicating TCP performance, creating possible IP layer fragmentation. Our proposed approach eases management, supports interoperability, and enhances privacy. It also supports reliability and scalability but with higher overhead. Simple Switch has been realised for completeness of the P4 standard [11], but not for performance. Currently, the implemented solution is facing two primary technical challenges. First related to Simple Switch, which is not a production-grade switch and secondly, the P4Runtime is also vulnerable to Man-in-the-Middle attacks and channel flooding [12]. Our proposed method can reduce/stop unwanted traffic analysis by a third party and stop flow tracing attacks. Technical cooperation from ISPs is highly desirable to transform such a proof-of-concept into a commercial-grade implementation.

V. Related Work

This work focuses on how user data packets can be controlled effectively via packet header marking. We found a large set of literature based on the solutions that provide source location privacy [13], privacy-aware outsourcing of storage and computation of sensitive online user data to public cloud platforms [14]. User data can be secured by implementing proper cryptographic tools/mechanisms. Li et al. proposed an approach to find the benign destination and execute a privacy-preserving verification process of the path [15]. Han et al. offer a source location protection protocol based on dynamic routing to counter the source location privacy problem [16]. In another work, Wang et al. proposed a multi-layer storage framework based on fog computing [17]. It uses the Hash-Solomon algorithm to protect online users' private data. Fan et al. suggested a privacy-preserving scheme against traffic analysis using network coding [18]. Jung et al. propose an anonymous, attribute-based privilege control scheme to address online user privacy issues [19] while Al-Muhtadi et al. propose a privacy-preserving protocol for ubiquitous computing environments [20]. In this work, we aim to let the sender control (at the switch level) how their data should travel to their destinations by encoding the preferences/rules with privacy markings at the switch level.

VI. Conclusion and Future Work

With the ubiquity of the Internet, privacy issues arise. The privacy of data flowing through networks is still a complex issue. In this paper, we presented our approach for *declaring user privacy requirements by marking packets and handling them according to their privacy markings*. Based on the P4 standard, our vendor-independent solution works with standard off-the-shelf hardware and open-source software. We showed initial results from a proof-of-concept in a cloud/fog research testbed. In future work, we aim to replace the software switch with a P4 hardware switch or a high-performance software implementation (such as Open vSwitch) to solve the scaling issue. We will extend the control plane to handle the flexible configuration of VLANs (such as automatically creating a new VLAN for new applications and removing unused ones) and VPNs (e.g. between a user's network and cloud providers). A GUI will be added to automate the rule configuration process and better interaction, where a user can define and monitor the privacy settings for different networks. Such a GUI should also provide different views based on the user's technical expertise.

Disclaimer

The countries and companies' names used in this paper are purely for research purposes and to make our scenarios applicable. No one should infer other meanings (directly or indirectly, explicitly or implicitly) from it.

References

[1] I. Sánchez-Rola *et al.*, "Can I Opt Out Yet? GDPR and the Global Illusion of Cookie Control," in *Proc. of the ACM Asia Conf. on Computer and Communications Security*, 2019.

[2] J. Su *et al.*, "De-Anonymizing Web Browsing Data with Social Networks," in *Proc. of the 26th Int'l Conf. on World Wide Web*, 2017.

[3] Y. Meng *et al.*, "Securing Consumer IoT in the Smart Home: Architecture, Challenges, and Countermeasures," *IEEE Wireless Communications*, vol. 25, 2018.

[4] P. Bosshart *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, 2014.

[5] A. Kuzmanović, "The Power of Explicit Congestion Notification," *ACM SIGCOMM Computer Communication Review*, vol. 35, Oct. 2005.

[6] S. Kent, "U.S. Department of Defense Security Options for the Internet Protocol," IETF, RFC 1108, Nov. 1991.

[7] E. G. Gran *et al.*, "NorNet Core – A Multi-Homed Research Testbed," *Computer Networks*, vol. 61, Mar. 2014.

[8] T. Dreibholz, "HiPerConTracer - A Versatile Tool for IP Connectivity Tracing in Multi-Path Setups," in *Proc. of the 28th IEEE Int'l Conf. on Software, Telecommunications and Computer Networks*, Sep. 2020.

[9] ——, "Evaluation and Optimisation of Multi-Path Transport using the Stream Control Transmission Protocol," Habilitation Treatise, University of Duisburg-Essen, Mar. 2012.

[10] T. Dreibholz and S. Mazumdar, "Find Out: How Do Your Data Packets Travel?" in *Proc. of the 18th IEEE Int'l Conf. on Network and Service Management*, 2022.

[11] P4 Language Consortium, *P4-16 Language Specification*, Vienna/Austria, May 2021.

[12] A.-A. Agape *et al.*, "Charting the Security Landscape of Programmable Dataplanes," *arXiv preprint arXiv:1807.00128*, 2018.

[13] M. Conti *et al.*, "Providing Source Location Privacy in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, 2013.

[14] J. Domingo-Ferrer *et al.*, "Privacy-Preserving Cloud Computing on Sensitive Data: A Survey of Methods, Products and Challenges," *Computer Communications*, vol. 140, 2019.

[15] T. Li *et al.*, "SRDPV: Secure Route Discovery and Privacy-Preserving Verification in MANETs," *Wireless Networks*, vol. 25, 2019.

[16] G. Han *et al.*, "A Source Location Protection Protocol based on Dynamic Routing in WSNs for the Social Internet of Things," *Future Generation Computer Systems*, vol. 82, 2018.

[17] T. Wang *et al.*, "A Three-Layer Privacy Preserving Cloud Storage Scheme Based on Computational Intelligence in Fog Computing," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, 2018.

[18] Y. Fan *et al.*, "An Efficient Privacy-Preserving Scheme against Traffic Analysis Attacks in Network Coding," in *Proc. of IEEE Int'l Conf. on Computer Communications*. IEEE, 2009.

[19] T. Jung *et al.*, "Privacy Preserving Cloud Data Access with Multi-Authorities," in *Proc. of IEEE Int'l Conf. on Computer Communications*. IEEE, 2013.

[20] J. Al-Muhtadi *et al.*, "Routing through the Mist: Privacy Preserving Communication in Ubiquitous Computing Environments," in *Proc. 22nd Int'l Conf. on Distributed Computing Systems*. IEEE, 2002.