P4-Protect: 1+1 Path Protection for P4

Steffen Lindner
University of Tübingen
steffen.lindner@uni-tuebingen.de

Daniel Merling
University of Tübingen
daniel.merling@uni-tuebingen.de

Marco Häberle
University of Tübingen
marco.haeberle@uni-tuebingen.de

Michael Menth
University of Tübingen
menth@uni-tuebingen.de

ABSTRACT

1+1 protection is a method to secure traffic between two nodes against failures in between. The sending node duplicates the traffic and forwards it over two disjoint paths. The receiving node assures that only a single copy of the traffic is further forwarded to its destination. In contrast to other protection schemes, this method prevents almost any packet loss in case of failures. 1+1 protection is usually applied on the optical layer, on Ethernet, or on MPLS.

In this work we propose the application of 1+1 for P4-based IP networks. We define an 1+1 protection header for that purpose. We describe the behavior of sending and receiving nodes and provide a P4-based implementation for the Behavioral Model version 2 (bmv2) software switch and the hardware switch Tofino Edgecore Wedge 100BF-32X. We illustrate how to secure traffic, e.g. individual TCP flows, on the Internet with this approach. Finally, we present performance results showing that the P4-based implementation efficiently works on the Tofino Edgecore Wedge 100BF-32X.

KEYWORDS

p4, software defined networking, 1+1 protection

ACM Reference Format:
Steffen Lindner, Daniel Merling, Marco Häberle, and Michael Menth. 2020. P4-Protect: 1+1 Path Protection for P4. In *3rd P4 Workshop in Europe (EuroP4'20), December 1, 2020, Barcelona, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3426744.3431327

1 INTRODUCTION

There are various concepts to secure traffic transmission against failure of path components such as links or nodes. The fastest is 1+1 protection. A sender duplicates traffic and forwards it over disjoint paths while the receiver forwards only the first copy received for every packet. In case of a failure, any packet loss can be avoided, which makes 1+1 protection attractive for highly reliable applications. 1+1 protection is implemented in optical networks to protect an entire trunk. It is also available for MPLS [10] and Ethernet [9], which are carrier technologies for IP and introduce signaling complexity. In this paper, we leverage the P4 programming language [3] to provide 1+1 protection for IP networks. We program P4 switches such that they feature IP forwarding, the sending and receiving node behaviour of 1+1 protection which includes IP encapsulation and decapsulation. We call this approach P4-Protect. Targets of our implementation are the software switch BMv2 and the hardware switch Tofino Edgecore Wedge 100BF-32X. A particular challenge is the selection of the fist copy of every duplicated packet at the receiver. We provide a controller that allows to set up 1+1 protection between P4 nodes implementing P4-Protect. Furthermore, protected flows can be added using a fine-granular description based on various header fields. We evaluate the performance of P4-Protect on the hardware switch. We show that P4-Protect can be used with only marginal throughput degradation and we illustrate that P4-Protect can significantly reduce jitter when both paths have similar delays.

The paper is structured as follows. Section 2 gives an overview of related work. Section 3 describes the 1+1 protection mechanism used for our implementation and extensions for its use on the general Internet. Section 5 presents a P4-based implementation including specifics for the Tofino Edgecore Wedge 100BF-32X. We evaluate the performance of P4-Protect on the hardware switch in Section 6 and conclude the paper in Section 7.

2 RELATED WORK

We review various resilience concepts for communication networks. Afterwards, we give examples for 1+1 protection.

2.1 Overview

Rerouting reorganizes the traffic forwarding to avoid failed components. This happens on a time scale of a second. Fast reroute (FRR) locally detects that a next hop is unreachable and deviates traffic to an alternative next hop [1]. The detection may take a few 10s of milliseconds so that traffic loss cannot be avoided. Both rerouting and FRR do not utilize backup resources under failure-free conditions, but their reaction time suffers from failure detection delay. 1:1 protection leverages a primary/backup path concept. To switch over, the head-end node of the paths needs to be informed about a failure, which imposes additional delay. With restoration, recovery paths may be dynamically allocated so that even more time is needed to establish the restoration paths [19, p. 31]. 1+1 protection duplicates traffic and sends it over two disjoint paths whereby the receiving node needs to eliminate duplicates. That method is fastest, but it requires extra capacities also under failure-free conditions. Some services can afford short network downtimes, other services greatly benefit from 1+1 protection's high reliability.

The surveys [15], [20], and [7] provide an overview of various protection and restoration schemes. The authors of [7] discuss survivability techniques for non-WDM networks like automatic protection switching (APS) and self healing rings (SHR) as well as dynamic restoration schemes in SONET. They further describe protection methods for optical WDM networks. A comprehensive overview of protection and restoration mechanisms for optical, SONET/SDH, IP, and MPLS networks can be found in [19].

SDN with inband signalling increases the need for fast and local protection against failures because the controller may no longer be reachable in case of a failure or highly loaded. In addition, with SDN new protection mechanisms can be implemented, e.g., to reduce state in the network. Examples are given in [14].

2.2 1+1 Protection

At first we will look at standards with respect to 1+1 protection, followed by other work related to 1+1 protection.

2.2.1 Standards. The ITU-T specification Y.1703 [10] defines a 1+1 path protection scheme for MPLS. It adds sequence numbers to packets and replicates them on disjoint paths. At the end of the paths, duplicate packets are identified by the sequence number and eliminated. P4-Protect works similarly. However, it does not require MPLS. It is compatible with IP and works over the Internet.

802.1CB [8] defines a redundant transmission mode for Time-Sensitive Networking (TSN), called *Frame Replication and Elimination for Reliability* (FRER). Each packet of a stream is equipped with a sequence number, replicated, and then sent through two disjoint paths to a destination. Both destination and/or traversing nodes eliminate duplicate packets. FRER supports two algorithms: *VectorRecoveryAlgorithm* and *MatchRecoveryAlgorithm*. With the *VectorRecoveryAlgorithm*, an acceptance window is used to accept packets with higher sequence numbers than expected. With the *MatchRecoveryAlgorithm* all sequence numbers except the last seen are accepted, which is used to prevent misbehaviour. In-order delivery is currently out of scope in 802.1CB.

DetNet [5] provides capabilities to carry data flows for real-time applications with extremely low data loss rates and bounded latency within a network. *Packet Replication and Elimination* (PRE) is a service protection method for DetNet, which leverages the 1+1 protection concept. PRE adds sequence numbers or time stamps to packets in order to identify duplicates. Packets are replicated and sent along multiple different paths, e.g., over explicit routes. Duplicates are eliminated, mostly at the edge of the DetNet domain. The *Packet Ordering Function* can be used at the elimination point to provide in-order delivery. However, this requires extra buffering.

2.2.2 Other Work on 1+1 Protection. The authors of [21] compare several implementation strategies of 1+1 protection, i.e, traditional 1+1 path protection, network redundancy 1+1 path protection (diversity coding) [2], and network-coded 1+1 path protection. Their analytical results show that diversity coding and network coding can be more cost-efficient, i.e., they require about 5-20% less reserved bandwidth. The delay impact of 1+1 path protection in MPLS networks has been investigated in [17]. McGettrick et. al [13] consider 10 Gb/s symmetric LR-PON. They reveal switch-over times to a backup OLT of less than 4 ms. Multicast traffic has often real-time requirements. Mohandespour et. al extend the idea of unicast 1+1 protection to protect multicast connections [16]. They formulate the problem of minimum cost multicast 1+1 protection as a 2-connectivity problem and propose heuristics. Braun et. al [4] propose maximally redundant trees for 1+1 protection in BIER, a stateless multicast transport mechanism. It leverages the concept of multicast-only FRR [11].

3 P4-PROTECT: CONCEPT

We first give an overview of P4-Protect. We present its protection header, the protection connection context, and the operation of the Protection Tunnel Ingress (PTI) and Protection Tunnel Egress (PTE).

3.1 Overview

With P4-Protect, a protection connection is established between two P4 switches. Protected traffic is duplicated by a PTI node and simultaneously carried through two protection tunnels to a PTE node. The PTE receives the duplicated traffic and forwards the first copy received for every packet.

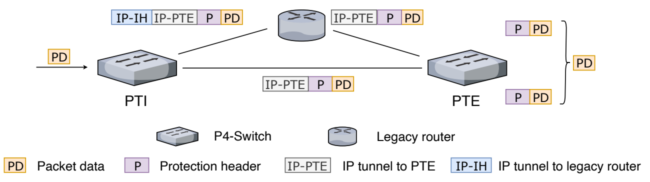

Figure 1: With P4-Protect, a PTI encapsulates and duplicates packets, and sends them over disjoint paths; the PTE decapsulates the packets and forwards only the first packet copy.

Figure 1 illustrates the protocol stack used with P4-Protect. The PTI adds to each packet received for a protected flow a protection header (P) that contains a sequence number which is incremented for each protected packet. The packet is equipped with an additional IP header (IP-PTE) with the PTE's IP address as destination. The PTI duplicates that packet and forwards the two copies over different paths. The paths may be different due to traffic engineering (TE) capabilities of the network or path diversity may be achieved through an additional intermediate hop. When the PTE receives a packet, it removes its outer IP header (IP-PTE). If the sequence number in the protection header is larger than the last sequence number received for this connection, it removes the protection header and forwards the packet; otherwise, the packet is dropped. The latter is needed as duplicate packets are also considered harmful.

3.2 Protection Header

The protection header contains a 24 bit Connection Identifier (CID), a 32 bit Sequence Number (SN) field, and an 8 bit *next protocol* field. The CID is used to uniquely identify a protection connection at the PTE. The sequence number is used at the PTE to identify duplicates. The *next protocol* field facilitates the parsing of the next header. We reuse the IP protocol numbers for this purpose.

3.3 Protection Connection Context

A protection connection is set up between a PTI and PTE. Their IP addresses are associated with this connection, including two interfaces over which duplicate packets are forwarded. For each connection, the PTI has a sequence number counter SN_{last}^{PTI} which is incremented for each packet forwarded over the respective protection connection. Likewise, the PTE has a variable SN_{last}^{PTE} which records the highest sequence number received for the respective protection connection. A CID is used to identify a connection at the PTE. A PTI may have several protection connections with the same CID but different PTEs (see Section 5.5.1).

3.4 PTI Operation

The PTI has a set of flow descriptors that are mapped to protection connections. If the PTI receives a packet which is matched by a specific flow descriptor, the PTI processes the packet using the corresponding protection connection. That is, it increments the SN_{last}^{PTI}, adds a protection header with CID, *next protocol* set to IPv4, and the SN set to SN_{last}^{PTI}. Then, an IP header is added using the PTI's IP address as source and the IP address of the PTE associated with the protection connection as destination. The packet is duplicated and forwarded over the two paths associated with the protection connection.

3.5 PTE Operation

During failure-free operation, the PTE receives duplicate packets via two protection tunnels. When the PTE receives a packet, it decapsulates the outer IP header. It uses the CID in the protection header to identify the protection connection and the corresponding SN_{last}^{PTE}. If the SN in the protection header is larger than SN_{last}^{PTE}, SN_{last}^{PTE} is updated by SN, the protection header is decapsulated, and the original packet is forwarded; otherwise, the packet is dropped.

The presented behavior works for unlimited sequence numbers. The limited size of the sequence number space makes the acceptance decision for a packet more complex. Then, a SN larger than SN_{last}^{PTE} may indicate a copy of a new packet, but it may also result from a very old packet. To solve this problem, we adopt the use of an acceptance window as proposed in [10]. The window is W sequence numbers large. Let SN_{max} be the maximum sequence number. If $SN_{last}^{PTE}+W < SN_{max}$ holds, a new sequence number SN is accepted if the following inequality holds:

$$SN_{last}^{PTE} < SN \leq SN_{last}^{PTE} + W \tag{1}$$

If $SN_{last}^{PTE} + W \geq SN_{max}$ holds, a new sequence number SN is accepted if one of the two following inequalities holds:

$$SN_{last}^{PTE} < SN \tag{2}$$

$$SN < SN_{last}^{PTE} + W - SN_{max} \tag{3}$$

This allows a packet copy to arrive $SN_{max} - W$ sequence numbers later than the corresponding first packet copy without being recognized as new packets.

AXE [12] tries to solve a similar problem, namely the de-duplication of packets. The hash of incoming packets is used to access special registers and associated header fields are stored. When another packet with the same hash arrives and the stored header fields match the incoming packet, the packet is a duplicate. No hash collision is considered. This technique detects duplicates quite reliably. However, AXE considers L2 flooding for learning bridges and therefore operates on relatively low bandwidths. P4-Protect must be able to de-duplicate several 100G connections, for the Tofino Edgecore Wedge 100BF-32X 3.2 Tb/s. Hence the AXE approach is not feasible due to the required register memory space and, depending on the hash algorithm, the high probability for hash collisions.

4 DISCUSSION

In this section the protection properties of P4-Protect are examined in more detail. Both, advantages and limitations of P4-Protect are discussed. The impact on jitter, packet loss and packet reordering are considered. To that end, we provide examples of traffic streams received by the PTE and their results after duplicate elimination.

4.1 Impact on Jitter

P4-Protect replicates packets to two preferable disjoint paths. If both paths suffer from jitter, P4-Protect can compensate the overall end-to-end jitter. Figure 2 illustrates the impact of P4-Protect on the overall end-to-end jitter. Packet 3 on path 1 has a very high delay due to jitter. As P4-Protect always forwards the first version of in-order packets, packet 3 is forwarded from the second path and thereby compensates the jitter delay.

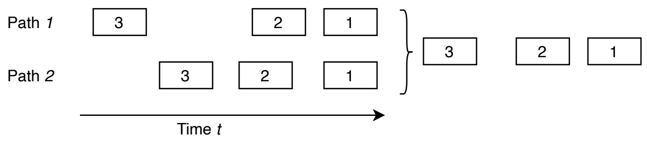

Figure 2: P4-Protect can reduce jitter.

4.2 Impact of Packet Loss

P4-Protect forwards the first version of a packet. If a path fails, all packet replicas of the other path are forwarded correctly. If individual packets are lost on one path, their replicas from the other path are not necessarily forwarded. This phenomenon ist illustrated in Figure 3. Four packets are replicated by the PTI and sent over two disjoint paths. The second path has a higher latency. As a result, packet 4 of the first path arrives before packet 3 on the second path. Now, SNE_{last}^{PTE} is set to 4, and as a consequence, packet 3 on the second path is discarded.

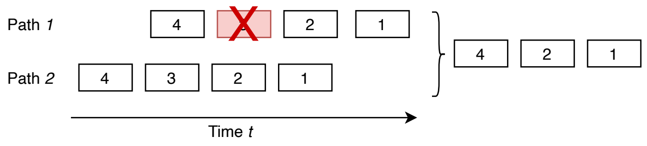

Figure 3: Packet loss may not be compensated by P4-Protect.

This behavior is due to the scalable design of P4-Protect. Only the last accepted sequence number is stored and checked at a new packet arrival. Missing packets are not memorized nor are packets buffered. This example clarifies that the objective of P4-Protect is to protect quickly against path failures, it is not to compensate for individual packet losses.

4.3 Packet Reordering

Packet reordering on a path has different sources, e.g., parallelism in network devices, link bundling, and special QoS configurations [18]. In case of packet reordering, P4-Protect may cause packet loss.

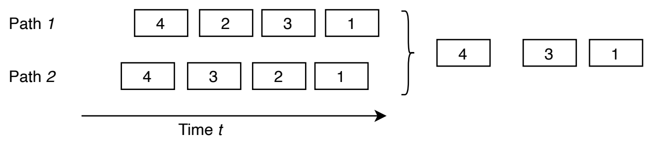

Figure 4: As it is not possible to check for lost packets, reordering leads to packet loss.

Figure 4 illustrates the impact of packet reordering. Path 1 has a slightly lower end-to-end latency than path 2. Due to packet reordering, the PTE receives the packets of path 1 in the order 1 3 2 4 instead of 1 2 3 4 . Moreover, packet 3 of the first path arrives slightly before packet 2 of the second path. As a result, the PTE accepts packets 1, 3 and 4 from path 1 and discards packet 2 from path 2. The main reason for this behavior is that P4-Protect does not memorize missing packets. Therefore, they cannot be accepted if they arrive in the wrong order. Limited arithmetic operations and storage access on our specific hardware target inhibit more sophisticated checks.

5 IMPLEMENTATION

In this section we present the implementation of P4-Protect. We describe the supported header stacks, explain the control blocks, their organization in ingress and egress control flow, and we reveal implementation details about some control blocks. Finally, we sketch most relevant aspects of the P4-Protect controller.

5.1 Supported Header Stacks

Incoming packets are parsed so that their header values can be accessed within the P4 pipeline. To that end, we define the following supported header stacks. Unprotected IP traffic has the structure IP/TP, i.e., IP header and some transport header (TCP/UDP), and protected IP traffic has the structure IP/P/IP/TP, i.e., the IP header with the PTE's address, the protection header, the original IP header, and a transport header. IP traffic without transport header is parsed only up to the IP header.

5.2 Control Blocks

We present three control blocks of our implementation of P4-Protect. They consider the packet processing by PTI and PTE.

5.2.1 Control Block: Protect&Forward. When the PTI receives an IP packet, it is parsed and matched against the Match+action (MAT) table ProtectedFlows. In case of a match, the packet is equipped with an appropriate header stack, duplicated, and sent to appropriate egress ports. In case of a miss, the packet is processed by a standard IPv4 forwarding procedure.

5.2.2 Control Block: Decaps-IP. When the PTE receives an IP packet with the PTE's own IP address, the IP header is decapsulated. If the next protocol indicates a protection header, the packet is handed over to the Decaps-P control block; otherwise, the packet is processed by the Protect&Forward control block since the resulting packet may need to be protected and forwarded.

5.2.3 Control Block: Decaps-P. In the Decaps-P control block, the PTE examines the protection header and decides whether to keep or drop the packet as it is a copy of an earlier received packet. To keep the packet, the protection header is decapsulated.

5.3 Ingress and Egress Control Flow

The inter-dependencies between the control blocks suggest the following ingress control flow: Decaps-IP, Decaps-P, Protect&Forward. At a mere PTI, no action is performed by the Decaps-IP and Decap-P control block. The Protect&Forward takes care that protected traffic is duplicated and sent over two different paths and that unprotected traffic is forwarded by normal IPv4 operation. At a mere PTE, protected traffic is decapsulated and selected before being forwarded by normal IPv4 operation. Unprotected traffic is just forwarded by normal IPv4 operation.

5.4 Control Block Implementations

In the following, we explain implementation details of the Protect&Forward control block and the Decaps-P control block. We omit the Decaps-IP control block as it is rather simple.

5.4.1 Protect&Forward Control Block. The operation of the Protect&Forward control block is illustrated in Figure 5. It utilizes the MAT ProtectedFlows to process all packets. It effects that protected traffic is encapsulated at the PTI with a protection header and an IP header for tunneling.

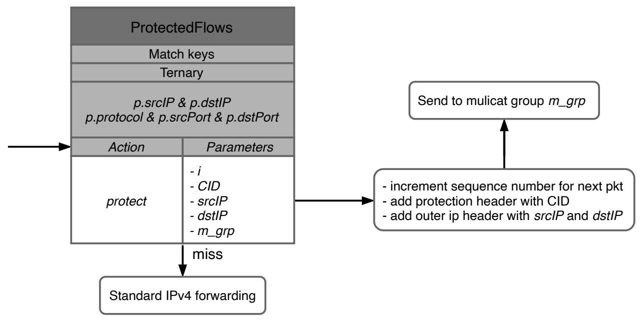
ProtectedFlows
Match keys
Ternary
p.srcIP & p.dstIP
p.protocol & p.srcPort & p.dstPort
Action
Parameters
protect
- i
- CID
- srcIP
- dstIP
- m_grp
miss
Standard IPv4 forwarding
- increment sequence number for next pkt
- add protection header with CID
- add outer ip header with srcIP and dstIP
Send to mulicat group m_grp

Figure 5: The MAT ProtectedFows inside the Protect&Forward control block is applied to IPv4 traffic.

The MAT ProtectedFlows uses a ternary match on the classic 5-tuple description of a flow: the source and destination IP address and port as well as the protocol field. In case of a match, the MAT maps a packet to a specifc protection connection and calls the protect action with the connection-specific parameters i, CID, $srcIP$, $dstIP$, and m_grp. The protect action increments the register $SN_{last}^{PTI}[i]$ where i is a connection-specific index to access a register containing the last sequence number. On the Tofino target, this is performed by a separate register action. The protect action further pushes a protection header on the packet including CID, i.e., the

CID, $SN^{PTI}_{last}[i]$, and the next protocol set to `IPv4`. Then, it pushes an IPv4 header with the IP address *srcIP* of the PTI as source IP and the IP address *dstIP* of the PTE as destination IP. The protocol field of this outer IP header is set to `P4-Protect`. Finally, the multicast group of the packet is set to *m_grp*. It is a connection-specific multicast group. It effects that the packet is duplicated and sent to two egress ports in order to deliver it via two protection tunnels to the PTE. In case of a miss, the packet is unprotected and handled by a standard IPv4 forwarding procedure, which is not further explained in this paper.

5.4.2 Decaps-P Control Block. The Decaps-P control block decides whether a packet is new and should be forwarded or dropped. It compares the sequence number SN of the packet's protection header with the last sequence number of the corresponding protection connection. The latter can be accessed by the register $SN^{PTE}_{last}[CID]$ where CID is given in the protection header. The acceptance is decided based on Equation (1) or Equation (3) depending on the value of SN and W where W is given as a constant.

As the check is rather complex, it requires careful implementation for the Tofino target [1]. It leverages the fact that we set $W = \frac{SN_{max}}{2}$. Furthermore, it requires a reformulation of Equation (1) and Equation (3).

If $W \leq SN$ holds, the following two inequalities must be met:

$$SN^{PTE}_{last} < SN \tag{4}$$
$$SN - SN^{PTE}_{last} \leq W \tag{5}$$

Otherwise, if $SN < W$, it is sufficient that only one of the following two inequalities holds:

$$SN^{PTE}_{last} < SN \tag{6}$$
$$W \leq SN^{PTE}_{last} - SN \tag{7}$$

Both cases are implemented as separate register actions on the Tofino target. With 32 bit sequence numbers, a minimum packet size of 40 bytes and a transmission speed of $C = 1$ Tb/s, a delay difference up to 1.6s can be compensated.

The bmv2 version of the implementation can be found at Github[2]. The Tofino version of the implementation can be found at Github[3] as well.

5.5 Controller for P4-Protect

P4-Protect's controller offers an interface for the management of protection connections and protected flows. It configures in particular the MAT ProtectedFlows but also other MATs needed for standard IPv4 forwarding or IP decapsulation. In the following, we explain the configuration of protection connections and protected flows.

5.5.1 Configuration of Protection Connections. A protection connection is established by choosing registers on PTI and PTE to record the last sequence numbers SN^{PTI}_{last} and SN^{PTE}_{last} of a protection connection. The connection identifier is the PTE's index to access SN^{PTE}_{last}. On the PTI, a different index i may be chosen to access SN^{PTI}_{last}. Furthermore, the registers are initialized with zero. Moreover, the controller sets up a multicast group *m_grp* for each connection so that its traffic will be replicated in an efficient way to the two desired interfaces.

5.5.2 Configuration of Protected Flows. A protected flow is established by adding a new flow rule in the MAT ProtectedFlows of the PTI. It contains an appropriate flow descriptor and the parameters to call the action protect. Those are the index i associated with the corresponding protection connection, the CID needed at the PTE to identify the protection connection, the IP address of the PTI, the IP of the PTE, and the multicast group *m_grp*.

6 EVALUATION

In this section we evaluate the performance of the implemented mechanism on the Tofino Edgecore Wedge 100BF-32X. First, we compare packet processing times with and without P4-Protect. Then, we demonstrate that very high data rates can be achieved with and without P4-Protect on a 100 Gb/s interface. Finally, we show that P4-Protect can provide a transmission service with reduced jitter compared to the jitter of both protection tunnels.

6.1 Packet Processing Time

P4-Protect induces forwarding complexity. To evaluate its impact, we leverage P4 metadata to calculate the time a packet takes from the beginning of the ingress pipeline to the beginning of the egress pipeline. This is sufficient for a comparison as all work for P4-Protect is done in the ingress pipeline and all considered forwarding schemes utilizes the same egress pipeline. We compare three forwarding modes: a plain IP forwarding implementation (plain), P4-Protect for unprotected traffic (unprotected), and P4-Protect for protected traffic (protected).

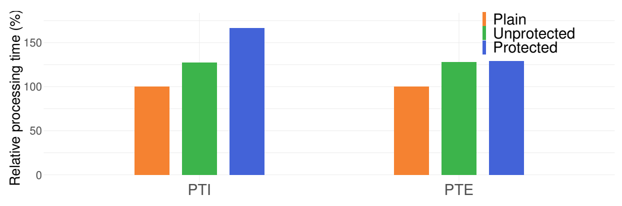

Figure 6: Ingress-to-egress packet processing time at PTI and PTE for three forwarding modes: plain, unprotected, and protected.

Figure 6 shows the ingress-to-egress packet processing time on both PTI and PTE for the three mentioned forwarding modes. The duration is given relative to the processing time for plain forwarding mode. We observe the lowest processing time at PTI and PTE for plain forwarding as it has the least complex pipeline. With P4-Protect, the processing time at both PTI and PTE is larger than with plain forwarding as the operations are more complex. At PTI, the processing time is even larger with protected forwarding (166%) than with unprotected forwarding (127%). At PTE, the processing times for protected and unprotected traffic are equal and 27% longer than with plain forwarding.

[1]Tofino is a high-performance chip which operates at 100 Gb/s so that only a limited set of operations can be performed for each packet, in particular in connection with register access.
[2]Repository: https://github.com/uni-tue-kn/p4-protect
[3]Repository: https://github.com/uni-tue-kn/p4-protect-tofino

In our implementations, we have used only a minimal IPv4 stack for all three forwarding modes. With a more comprehensive IPv4 stack, the relative overhead through P4-Protect is likely to be smaller.

6.2 TCP Goodput

We set up iperf3 connections between client/server pairs and measure their goodput. Each iperf3 connection consists of 15 parallel TCP flows. Two switches are bidirectionally connected via two 100 Gb/s interfaces. Four client/server pairs are connected to the switches via 100 Gb/s interfaces. Up to 4 clients download traffic from their servers via the trunk between the switches.

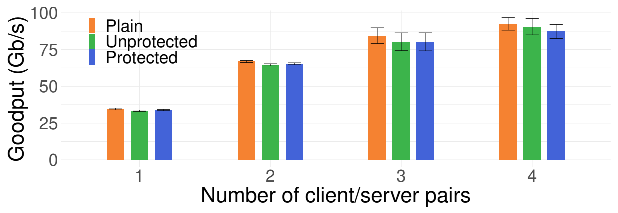

Figure 7: Impact of varying number of client/server pairs exchanging traffic with iperf3.

Figure 7 shows the overall goodput for a various number of client/server pairs, each transmitting traffic over a single TCP connection. The goodput is given for the forwarding modes plain, unprotected, and protected. We performed 20 runs per experiment and provide the 95% confidence interval.

A single, two, and, three TCP connections cannot generate sufficient traffic to fill the 100 Gb/s bottleneck link. However, with four TCP connections a goodput of around 90 Gb/s is achieved. This is less than 100% because of overhead due to Ethernet, IP, and TCP headers and due to the inability of TCP to efficiently utilize available capacity at high data rates. Most important is the observation that all three forwarding modes lead to almost identical goodput. The goodput for protected and unprotected forwarding is slightly lower than plain forwarding, which is apparently due to the operational overhead of P4-Protect.

6.3 Impact on Jitter

We examine the effect of 1+1 path protection on jitter. Two hosts are connected to two Tofino Edgecore Wedges 100BF-32X. The switches are connected with each other via two paths with intermediate Linux servers. Their interfaces are bridged and cause an artificial, adjustable, uniformly distributed jitter. We leverage the *tc* tool for this purpose [6]. All lines have a capacity of 100 Gb/s.

In our experiment, we send pings between the two hosts with and without P4-Protect. Figure 8 reports the average round trip time (RTT) deviation for the pings. Unprotected traffic suffers from all the jitter induced on a single path. Protected traffic suffers only from about half the jitter. This is because P4-Protect forwards the earliest received packet copy and minimizes packet delay occurred on both links.

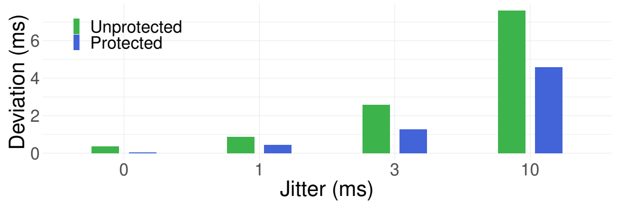

Figure 8: Impact of 1+1 protection on jitter.

7 CONCLUSION

In this paper we proposed P4-Protect for 1+1 path protection with P4. It may be utilized to protect traffic via two largely disjoint paths. We presented an implementation for the software switch bmv2 and the hardware switch Tofino Edgecore Wedge 100Bf-32X. The evaluation of P4-Protect on the hardware switch revealed that P4-Protect increases packet processing times only little, that high throughput can be achieved with P4-Protect, and that jitter is reduced by P4-Protect when traffic is carried over two path with similar delay but large jitter.

ACKNOWLEDGMENTS

The authors acknowledge the funding by the Deutsche Forschungsgemeinschaft (DFG) under grant ME2727/1-2. The authors alone are responsible for the content of the paper.

REFERENCES

[1] A. Atlas et al. 2008. RFC5286: Basic Specification for IP Fast Reroute: Loop-Free Alternates .
[2] Ender Ayanoglu et al. 1993. Diversity coding for transparent self-healing and fault-tolerant communication networks. *IEEE ToC* 41(11) (1993).
[3] P. Bosshart et al. 2014. P4: Programming Protocol-Independent Packet Processors. *ACM CCR* 44(3) (2014).
[4] Wolfgang Braun et al. 2017. Performance Comparison of Resilience Mechanisms for Stateless Multicast Using BIER. In *IFIP/IEEE.*
[5] Norman Finn, Pascal Thubert, Balazs Varga, and János Farkas. 2019. Deterministic Networking Architecture. RFC 8655. https://doi.org/10.17487/RFC8655
[6] Linux Foundation. 2019. *Linux Traffic Control.*
[7] A. Fumagalli et al. 2000. IP restoration vs. WDM protection: is there an optimal choice? *IEEE Network Magazine* 14(6) (2000).
[8] IEEE Computer Society. 2017. *Frame Replication and Elimination for Reliability.* Technical Report.
[9] ITU. 2006. ITU-T Recommendation G.803/Y.1342 (2006), Ethernet Protection Switching .
[10] ITU. 2010. ITU-T Recommendation G.7712/Y.1703 (2010), Internet protocol aspects – Operation, administration and maintenance.
[11] A. Karan et al. 2015. RFC7431: Multicast-Only Fast Reroute.
[12] James McCauley, Mingjie Zhao, Ethan J. Jackson, Barath Raghavan, Sylvia Ratnasamy, and Scott Shenker. 2016. The Deforestation of L2. In *Proceedings of the 2016 ACM SIGCOMM Conference.*
[13] Sèamas McGettrick et al. 2013. Ultra-fast 1+1 protection in 10 Gb/s symmetric Long Reach PON. In *IEEE ECOC).*
[14] Daniel Merling et al. 2018. Efficient Data Plane Protection for SDN. IEEE (NetSoft).
[15] Christopher Metz. 2000. IP protection and restoration. *IEEE Internet Computing* 4(2) (2000).
[16] Mirzad Mohandespour et al. 2015. Multicast 1+1 protection: The case for simple network coding. In *IEEE ICNC.*
[17] Grazziela Niculescu et al. 2010. The Packet Delay in a MPLS Network Using "1+1 Protection. In *IEEE Advanced International Conference on Telecommunications.*
[18] Michal Przybylski, Bartosz Belter, and Artur Binczewski. 2005. Shall we worry about Packet Reordering? *Computational Methods in Science and Technology* 11.
[19] Jean Philippe Vasseur et al. 2004. *Network Recovery.* Morgan Kaufmann.
[20] Dongyun Zhou et al. 2000. Survivability in Optical Networks. *IEEE Network Magazine* 14(6) (2000).
[21] Harald Øverby et. al. 2012. Cost comparison of 1+1 path protection schemes: A case for coding. In *IEEE ICC.*