Computer Networks 127 (2017) 233–242

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Flow termination signaling in the centralized pre-congestion notification architecture

Frank Wetzels[a,*], Hans van den Berg[a,b,d], Joost Bosman[a], Rob van der Mei[a,c]

[a] Centrum Wiskunde en Informatica, Amsterdam, The Netherlands
[b] TNO, The Hague, The Netherlands
[c] Vrije Universiteit Amsterdam, The Netherlands
[d] Technische Universiteit Twente, Enschede, The Netherlands

ARTICLE INFO

Article history:
Received 14 February 2017
Revised 9 July 2017
Accepted 14 August 2017

Keywords:
Centralized decision
Point pre-congestion notification
Signaling
Software defined networking
Flow termination

ABSTRACT

Pre-congestion notification (PCN) protects inelastic traffic by using feedback on network link loads on and acting upon this accordingly. These actions comprise to admission control and termination of flows. Two PCN architectures have been defined by IETF: the centralized and decentralized PCN architecture. The decentralized PCN architecture has received much attention in the literature whereas the centralized PCN architecture has not. In the decentralized architecture, feedback is sent from the egress nodes to ingress nodes, which then take and apply decisions regarding admission of new flows and/or termination of ongoing flows. Signaling occurs only between ingress and egress nodes.

In the centralized architecture these decisions are made at a central node, which requires proper signaling for action and information exchange between the central node and the egress and ingress nodes. This signaling has been suggested by other authors, but is not fully defined yet. Our contribution is twofold. We define signaling in the centralized PCN architecture focussing on flow termination, which completes the definition of the signaling in the centralized PCN architecture. Secondly, we run extensive simulations showing that the proposed signaling works well and that the performances of the centralized PCN and the decentralized PCN architectures are similar. Hence, it is expected that results from existing research on the effectiveness of decentralized PCN are also valid when the centralized PCN architecture is used.

1. Introduction

Currently, video and web traffic are major contributors to internet traffic. Web traffic is built upon an elastic transport protocol, mostly TCP which can adapt to congestion. Nowadays, also (non-real-time) video traffic like YouTube is increasingly delivered over TCP, which requires the video coding to be able to adapt in case of congestion. However, real-time video applications and VoIP use an inelastic protocol (e.g. UDP). Such protocol cannot adapt to congestion in the network and may suffer by packet loss, increased delay, greater jitter and reduced available bandwidth. This affects real-time applications like VoIP, VoD, IPTV and others. Which leads to a degradation of the quality of service (QoS) experienced by the users of real-time applications.

Pre-congestion notification (PCN) protects inelastic traffic by flow admission and flow termination [1] when certain criteria related to the network load are met [2,3]. Decisions to take actions are based on traffic measurements in the network and reporting upon these measurements. Traffic enters the PCN-domain at an ingress-node and leaves at an egress-node. While traffic flows through the network, passing internal nodes, traffic is classified against pre-defined PCN-related thresholds. Based on the amount of PCN marked traffic [4] a report is created at fixed time periods and sent to the decision making node. These reports may trigger admission control and flow termination decisions. When traffic leaves the network, marked traffic is administered for the next report to be sent. PCN can be applied in a centralized and decentralized architecture. In this paper we denote by cPCN and dPCN, the centralized and *de*centralized PCN architecture respectively. In dPCN, all egress-nodes send feedback to *ingress-nodes* which take and apply decisions on flows. In cPCN, all egress-nodes send feedback to a *central node*, the decision point (DP), which decides what to do upon such feedback.

After a decision is made, the ingress nodes need to get instructed what to do: Admit or block a new flow, i.e. admission control (AC), or terminate one or more existing flows, i.e. flow termination (FT). The signaling between the DP and the ingress-

* Corresponding author.
E-mail address: frank.wetzels@cwi.nl (F. Wetzels).

http://dx.doi.org/10.1016/j.comnet.2017.08.010
1389-1286/

nodes has been suggested by other authors [5,6]. However, some essential components are missing in the signaling. This paper fills in the current gaps in cPCN signaling. In addition, extensive simulations have been carried out for both cPCN and dPCN as well as for a network without PCN in order to show the effectiveness of our proposed signaling. These simulations show that the proposed cPCN signaling works properly from a functional point of view, and that the performances of the cPCN and dPCN architectures are very similar. Hence, it is expected that results from existing research on the effectiveness of dPCN are also valid when cPCN is used. As in the aforementioned references [1–6], our specifications and simulations are based on 'traditional' networks assuming an interior gateway protocol and destination based forwarding. However, the cPCN signaling architecture fits very well to the centralized nature of the control architecture of emerging Software Defined Networks (SDN, see e.g. [7–9]) that (amongst others) takes care of flow routing in the data plane. Therefore, the outcome of our study also shows potential for enriching SDN with flow admission control and flow termination functionalities according to the cPCN approach. To the best of our knowledge such an extension of SDN has not yet been considered in the literature. The remainder of this paper is organized as follows. We start with a background on PCN and related work in Section 2. Section 3 highlights the proposed changes and additions to the signaling required in the cPCN. Section 4 describes these signaling modifications and additions in great detail for both admission control and flow termination. At the end of the section the identifiers and messages are defined in detail. In Section 5 the results of the simulations done in networks with cPCN, dPCN and without PCN are presented and discussed. Finally, discussions, conclusions as well as topics for future work are given in Section 6.

2. Background

The general architecture of PCN is given in [1]. If a new flow requests to enter the PCN-domain, it is decided whether or not this flow gets admitted to the PCN-domain (AC). This decision is based on the traffic load in the network. If an unusual event occurs in the network, for example a link failure, traffic gets rerouted and severe traffic overload on one or more links may happen. In such cases PCN may even decide to terminate one or more existing (previously admitted) flows (FT). The decision point (DP) decides whether a new flow gets admitted or blocked and what flows should be terminated, if applicable. In dPCN, each ingress node acts as DP for associated traffic, i.e. no central DP exists. In the cPCN, one node acts as DP. The DP does not take part in the data forwarding. The decision criteria for AC and FT are specified in [2,3] for the single marking (SM) and controlled load (CL) implementation respectively. In this paper we will focus on the signaling of the CL implementation in cPCN with one DP.

A brief overview of the research done on PCN is given below. In [10–14] the effectiveness of PCN is investigated in the context of a network with CBR traffic with on-off periods approximating different types of voice and video traffic. In particular, in [14] different PCN-based AC algorithms are considered and compared under various network load conditions. Reference [13] proposes a new measurement algorithm (sliding window) for AC based on bandwidth metering. In [15] an autonomous AC algorithm is proposed optimized for bursty traffic, which adapts itself based on previous measurements. Performance and parameter sensitivity analysis is done in [16] for both the SM and CL in dPCN. In [17] an summary is given of many aspects of PCN including the working, benefits, signaling and limitations of PCN in general.

We will now focus on the signaling in cPCN, in particular the associated signaling aspects. To determine whether AC and/or FT is required, the DP needs feedback from the egress nodes. The feedback is generated per aggregate at fixed time intervals by egress nodes and sent to the DP. An ingress-egress-aggregate, aggregate in short, is a set of flows which travel in the network from an ingress node to an egress node. The DP needs to exchange information with ingress nodes as to what the actual aggregate rate is, inform on whether to admit or block flows and to inform the ingress node(s) which ongoing flow(s) need to be terminated, if the FT criterium is met. The egress nodes need to send feedback to the DP which should contain information on the load per aggregate.

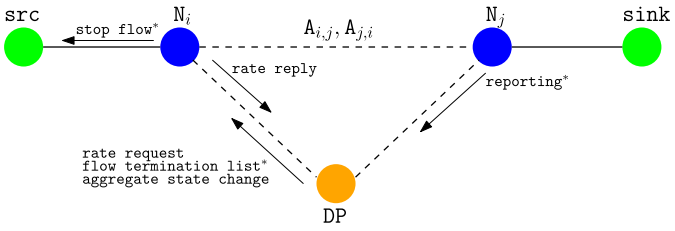

Fig. 1. Signaling data flow in cPCN. An '*' indicates a change to the current definition or a new definition.

On the signaling in a PCN-domain, P. Eardley [1] refers to related work that consider specific signaling protocols or frameworks like next steps in signaling (NSIS, [18]), resource reservation protocol (RSVP, [19]) and extensions to RSVP [20]. In [2], signaling is considered out of scope and refers to [20] as well. NSIS mainly focuses on protocols for signaling that follow the same paths along which the user-data flows, i.e. path-coupled signaling. NSIS considers the path-*de*coupled signaling briefly. In SDN and cPCN, all signaling is decoupled from the data path since all signaling happens between SDN switches and the SDN controller. In [5], requirements for signaling in a PCN-domain are described. Karagiannis et al. [5] restricts to feedback signaling between egress-nodes and DP and the signaling between DP and ingress-node on the aggregate-rate request. The signaling between DP and ingress-nodes on which flows to terminate and how to stop a source from sending a current (to be terminated) flow is not specified. For that, a reference is made to the common open policy service architecture (COPS, [21]) and the diameter based protocol (DBP, [22]) as a basis for a full signaling architecture. In [6] a signaling protocol, regular-check-based flow termination (RCFT), is proposed using RSVP as a carrier. It fills in the gap in the FT-communication between egress and ingress nodes. However, RCFT is focused on dPCN. In [17] the path-decoupled signaling in cPCN is discussed. However, it does not define the actual signaling in case of termination of flows. In this paper, we will propose signaling in case of flow termination and make an addition to the reporting. Simulation is used to check the functional correctness of these extensions and evaluate their performance.

3. Signaling in the cPCN

In this section the signaling between ingress-egress nodes, ie-nodes in short, and DP is considered, i.e. PCN signaling in the cPCN. The following components will be introduced: the *flow-rate*, the *flow-termination list* and the *flow-off* signal.

Refer to Fig. 1. The focus will be on two ie-nodes and one DP. This small network with one DP is no restriction as for every edge-node in the network the signaling below still applies. Considering multiple DPs would introduce other issues, like synchronization between DPs and the placement of DPs as well. These issues would distract our focus from the signaling. Between the ie-nodes two unidirectional aggregates exist. By $A_{i,j}$, we refer to the unidirectional aggregate from ie-node N_i to ie-node N_j The DP will not be part of any data-path, i.e. no aggregate will flow through the DP.

Fig. 1 gives an overview of the flow of the messages which includes flow termination signaling. Section 4 gives a detailed description of the signaling. We will give a brief summary of the messages involved:

Reporting: Egress-nodes send a report to the DP through the network at regular intervals. A report contains the NM_{rate}, ThM_{rate} and ETM_{rate}, the amount of traffic which is NM, ThM or ETM marked as per [4], per aggregate in bytes per second. We added the flow-rate.

Rate request: If required, the DP will request the aggregate ingress-rate from an ingress-node. A rate-request contains the PCN_{rate}, the total traffic entering an aggregate, in bytes per second.

Flow termination list: If required, the DP will inform an ingress-node which flows should be terminated. This list is defined in this paper.

Aggregate state change: If required, the DP will inform an ingress-node to change the state of an aggregate.

Stop flow: If required, an ingress-node informs the source to stop the flow.

With the '→' we highlight a change or additional definition to the signaling or data-object used within the signaling. By PCN_{rate} we denote the amount of traffic that enters an ingress node destined to a certain aggregate. With NM_{rate}, ThM_{rate} and ETM_{rate} we denote the amount of traffic that is either not-marked (NM), threshold-marked (ThM) and excess-traffic-marked (ETM). The marking occurs in the ECN bits in the TOS byte in the IP header as per [4]. When packets leave the network, the rate of NM, ThM and ETM marked traffic is reported to the DP. By CLE and CLE_{lim} we denote the congestion level estimator and the CLE threshold upon which AC takes place respectively. The CLE is given by,

$$CLE = \frac{ThM_{rate} + ETM_{rate}}{NM_{rate} + ThM_{rate} + ETM_{rate}}. \quad (1)$$

If the denominator equals to zero, the CLE is defined as zero. The CLE_{lim} is configured at the DP. Each CLE belonging to an aggregate is compared to the CLE_{lim} upon which the a decision is taken, i.e. admit, block or terminate flows. The bandwidth up-to which AC would admit new flows is represented by CLE_{lim} and called the admissible rate (ADM_{rate}).

Below the signaling between ie-nodes and the DP is explained. Assume that the traffic is flowing through the PCN-domain, i.e. a certain number of flows is admitted to the PCN-domain and the end of a reporting period is about to take place. The following events occur.

1. Egress-nodes send reports to the DP. The bandwidth consumption is kept as low as possible by combining reports for multiple aggregates per feedback. A report contains an aggregate identification, NM_{rate}, ThM_{rate}, ETM_{rate} and optionally the CLE. As per [5], an egress-node may include flow-identifiers that represent flows that experienced ETM-traffic when sending a report to the DP. However, the flow-rate, i.e. the number of bytes sent by a flow, is not defined as part of such report. The flow-rate is required in order to determine the amount of traffic that needs to be terminated.
   → For each flow-identifier, the *flow-rate* is added to the report.
2. A report triggers admission or blocking of new flows. The AC-criterium is defined as follows:
   (a) If a report shows CLE $>$ CLE_{lim} then upon the start of the next reporting period the aggregate associated to this report is set to or stays at BLOCK state. No new flows are admitted during the next reporting period.
   (b) If a report shows CLE $\leq$ CLE_{lim} then upon the start of the next reporting period the aggregate associated to this report is set to or stays at ADMIT state. New flows are admitted during the next reporting period.
   Note that the event of CLE $=$ CLE_{lim} is expected not to happen since these are continuous values and $CLE_{lim} > 0$.
3. A report *may* lead to FT in an aggregate. FT takes place if both of the following apply:
   (a) A report shows CLE $>$ CLE_{lim} and $ETM_{rate} > 0$.
   The DP requests the PCN_{rate} from the ingress node by sending a rate-request.
   (b) The next consecutive report (concerning the same aggregate as before) shows CLE $>$ CLE_{lim} and $ETM_{rate} > 0$.
4. As soon as the aggregate flow-rate is received by the DP, the DP can determine the number of flows to be terminated. From the set of flows that contain ETM-traffic, flows are chosen at random. As per [2], the amount of traffic to be terminated equals to PCN_{rate} - NM_{rate} - ThM_{rate}.
5. The DP informs the ingress-node which flows need to be terminated for each aggregate starting at that ingress-node. Therefore, the DP sends a list of aggregates each containing a list of flows to be terminated.
   → A *flow-termination list* will be defined.
6. Based on the information received from the DP, the ingress-node informs associated source(s) to stop the flow.
   → A *flow-off* signal will be defined.

We conclude this section with the following remarks:

Remark 1. A source starts a flow without sending a start-message. This is not a restriction as a first packet indicates that a flow starts.

Remark 2. The number of flows to be terminated depends on the size of individual flows and the amount of ETM_{rate} that was reported by the egress-nodes. From the set of flows that experienced ETM traffic, flows are chosen at random. The ETM_{rate} recorded by the egress-node will never be greater than the sum of the rates of all flows that showed excess-marked packets. Indeed, if a flow shows excess-marked packets, then this flow will also show packets that are not-marked (in PCN sense) and threshold-marked. The latter two are not part of the excess-rate.

4. Decision point and ingress nodes signaling – a detailed description

In this section the specific signaling between a decision point and ie-nodes is considered. By using a `teletype font`, we will denote an instantiation of a parameter or object in our networks, simulations and signaling. Without loss of generality, we can focus on the network shown in Fig. 1 whereby PCN is considered to be configured properly on all links occurring in the network except on the links that connect the DP. Two ie-nodes (N_i and N_j), a DP (DP), a source (`src`) and a sink (`sink`) are shown. In this network `src` sends data to `sink`. Two aggregates exist: $A_{i,j}$ for traffic from `src` to `sink` and $A_{j,i}$ for traffic from `sink` to `src`. The sink will discard any received traffic. Therefore, aggregate $A_{j,i}$ will not contain any flows. This is no restriction as an one-way traffic flow sufficiently illustrates our signaling. By $a_{i,j}$, we denote the identifier of aggregate $A_{i,j}$ used in the signaling. By f_i, we refer to the identification of flow F_i. The dashed lines in the network indicate a logical connection, i.e. the nodes may be directly connected or internal nodes exist between them. Traffic between all nodes is routed based on the shortest path first algorithm without multi-path routing. This is not a restriction for the signaling proposed in this paper. An aggregate is an unidirectional entity that represents flows running from a common ingress-node to a common egress-node. In reality, multiple aggregates exist and may run through common circuits. Two (or more) aggregates running through a common circuit influence

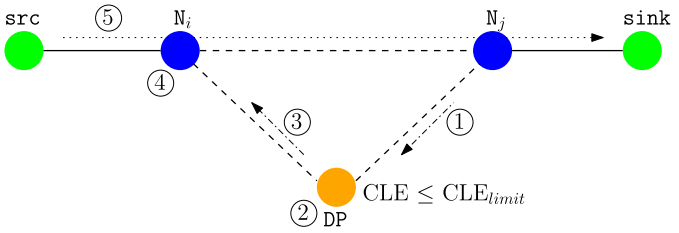

Fig. 2. Signaling in cPCN for flow admission whereby $A_{i,j}$ is in ADMIT state before data arrives at N_i.

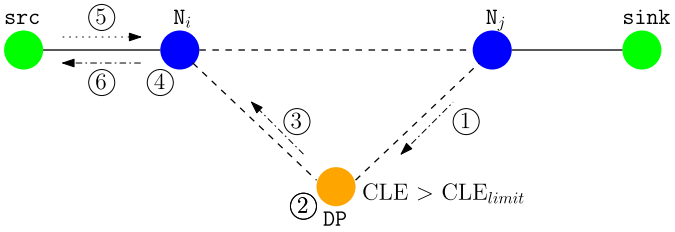

Fig. 3. Signaling in cPCN for flow admission whereby $A_{i,j}$ is in BLOCK state before data arrives at N_i.

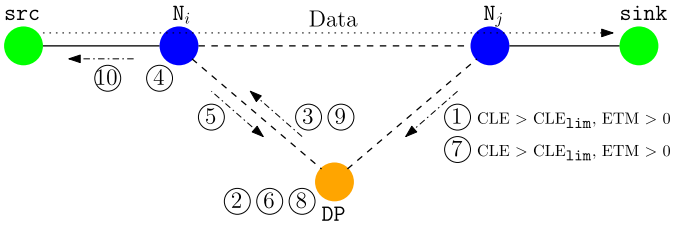

Fig. 4. Signaling in cPCN for flow termination.

their load. However, that does not affect the *signaling* between DP and ingress- and egress-nodes.

The following section describes in great detail the signaling required for cPCN to operate properly. Three situations are distinguished, two during admission control and one situation during flow termination. In Sections 4.1.1 and 4.1.2, the signaling is considered while an aggregate is in ADMIT and in BLOCK state respectively.

The signaling during flow termination is considered in Section 4.2. Section 4.3 gives a summary of the data objects used in the signaling. Note that all signaling starts at the end of each reporting period.

4.1. Admission control signaling

During admission control, two situations can occur. A new flow gets admitted or does not get admitted, i.e. gets blocked. Below these two situations are considered.

4.1.1. ADMIT state

Refer to Fig. 2. After aggregate $A_{i,j}$ is put in ADMIT state, a new flow F comes in. The following events take place whereby the sequence of numbers represent the sequence of occurrence of the corresponding events in time. Arrows indicate the direction of the associated signal or data flow.

(1) At the end of a reporting period, N_j sends a report to DP concerning $A_{i,j}$. This report contains the NM_{rate}, ThM_{rate}, ETM_{rate} and CLE. It also contains the flow-rates of each individual flows flowing through $A_{i,j}$. The use of flow-rates will be covered in Section 4.2.
(2) The reported values are recorded in a local database. If the report implies $CLE \leq CLE_{lim}$, than $A_{i,j}$ stays or changes to ADMIT state. If no aggregate state change should happen, no message is sent. If a state change should happen, then a STATECHANGE message is sent to N_i. Assume an aggregate state change to ADMIT should happen. The DP records the new state in its local database.
(3) DP sends a STATECHANGE($a_{i,j}$, ADMIT) to N_i.
(4) N_i changes the state of $A_{i,j}$ to ADMIT and records it in its local database.
N_i continues measuring the flow-rate of existing flows and continues measuring the flow-rate into $A_{i,j}$.
(5) F arriving at N_i will be admitted to the PCN-domain for transport. F is added to $A_{i,j}$ and N_i starts measuring the flow-rate of F. At the end of the new reporting period, the F is included in the reporting by N_j.

Note that if no state change of $A_{i,j}$ should happen, the above signaling is restricted to sending reports from N_j to DP only.

4.1.2. BLOCK state

Refer to Fig. 3. A flow F starts while aggregate $A_{i,j}$ is in BLOCK state. Existing, previously admitted flows, are not affected by a state change of an aggregate to BLOCK state. These flows are not considered. The following events happen.

(1) At the end of a reporting period, N_j sends a report to DP concerning $A_{i,j}$. It contains the NM_{rate}, ThM_{rate}, ETM_{rate} and CLE. This report also contains the flow-rates of each individual flows flowing through $A_{i,j}$. The use of flow-rates will be covered in Section 4.2.
(2) The reported values are recorded in the local database at the DP. If the report implies $CLE > CLE_{lim}$, than $A_{i,j}$ stays or changes to BLOCK state. If no aggregate state change should happen, no message is sent. If a state change should happen, then a STATECHANGE message is sent to N_i.
$ETM_{rate} = 0$, otherwise the FT criterium is met and the situation in Section 4.2 applies.
The DP records the new state in its local database.
(3) DP sends STATECHANGE($a_{i,j}$, BLOCK) to N_i.
(4) N_i changes the state of $A_{i,j}$ to BLOCK and records it in its local database.
N_i continues measuring the flow-rate of existing flows and continues measuring the flow-rate into $A_{i,j}$. F is not admitted.
(5) Between receiving the STATECHANGE and sending the FLOWOFF message, the first and all subsequent packets of F are dropped by N_i until F stops.
(6) N_i sends a FLOWOFF message to src. F stops.
The assumption is that source determines a new starting time and retries. If src retries, the above admission process is restarted.

Note that while $A_{i,j}$ is in BLOCK state and no new flows arrive at N_i, events (5) and (6) do not occur.

4.2. Flow termination signaling

Admission control and flow termination may be applied to the network independently. We assume that both are active in the network. Define r_n as the n-th report sent by N_j since start. Refer to Fig. 4. The situation in the network is at follows. A number of flows has been admitted in the past. Aggregate $A_{i,j}$ is either in ADMIT or in BLOCK state. The following events happen.

(1) At the end of a reporting period, N_j sends report r_n to the DP concerning $A_{i,j}$. This report contains the NM_{rate}, ThM_{rate}, ETM_{rate} and CLE. It also contains the flow-rates of each individual flows flowing through $A_{i,j}$.

(2) The reported values are recorded in the local database at the DP. Assume CLE > CLE_{lim}. $\text{A}_{i,j}$ stays or changes to BLOCK state. If no aggregate state change should happen, no message is sent. If a state change should happen, then a STATECHANGE message is sent to N_i.
Assume a state-change is required.
The DP records the new state in its local database.
If $\text{ETM}_{\text{rate}} = 0$ then the FT criterium is not met and the situation in Section 4.1.2 applies.
Assume $\text{ETM}_{\text{rate}} > 0$ then DP requires the ingress-rate of $\text{A}_{i,j}$.
(3) DP sends a STATECHANGE($\text{a}_{i,j}$, BLOCK) to N_i.
DP sends a RATEREQUEST($\text{a}_{i,j}$) to N_i.
(4) N_i changes the state of $\text{A}_{i,j}$ to BLOCK and records it in its local database.
N_i continues measuring the flow-rate of existing flows and continues measuring the ingress-rate into $\text{A}_{i,j}$.
N_i determines the ingress-rate of $\text{A}_{i,j}$ during the current reporting period.
(5) N_i sends a RATEREPLY($\text{a}_{i,j}$, PCN_{rate}) to DP.
(6) DP records the ingress-rate for $\text{A}_{i,j}$ and records the fact that a rate-reply has been received for $\text{A}_{i,j}$.
(7) At the end of the (next) reporting period, N_j sends report r_{n+1} to the DP concerning $\text{A}_{i,j}$.
(8) The reported values are recorded in the local database at the DP. Assume CLE > CLE_{lim}. $\text{A}_{i,j}$ stays in BLOCK state. No STATECHANGE will be sent.
If $\text{ETM}_{\text{rate}} = 0$ then the FT criterium is not met and the situation in Section 4.1.2 applies.
If $\text{ETM}_{\text{rate}} > 0$ and a RATEREPLY was received for $\text{A}_{i,j}$ previously as a result of r_n (event (6)), DP will determine the amount of traffic that needs to be terminated.
As a consequence, from the set of admitted flows flowing through $\text{A}_{i,j}$ that experienced ETM-traffic, flows are chosen randomly matching up the amount of traffic that needs to be terminated. For this, the recorded flow-rates that were sent in the last report are used.
A flow termination list needs to be sent. The flow termination list contains a list of aggregates, A_{i,j_1}, A_{i,j_2}, ... each with a list of admitted flows. Note that all aggregates in a termination list source from the same ingress node N_i.
(9) DP sends a FTLIST((a_{i,j_1}, f_1, ..., f_{l_1}), (a_{i,j_2}, f_1, ..., f_{l_2}), ...) to N_i.
(10) For each flow listed in the FTLIST, N_i records the flow as terminated and sends a FLOWOFF message to the corresponding source.

4.3. Summary of proposed data objects

This section gives a list of the definitions of the data objects that are used in the proposed signaling in the previous sections. Note that it is assumed that sources and destinations of flows connect to one PCN-domain. As a consequence, we also assume no network address translation takes place while packets travel from source to destination.

Flow-identifiers. The use of source and destination addresses and, if needed, a protocol identifier of a flow are sufficient to identify a flow. The assumption is that inside a PCN-domain, no network address translation or proxy-service takes place that would affect any flow. As a consequence, the ie-nodes can identify the flows based on their source and destination addresses. Therefore, a flow identifier could consist of a 2-tuple (`src-address, dst-address`) or a 3-tuple (`src-address, dst-address, protocol`).
Any feedback sent from egress-node to a DP, will lead to the DP learning about the current flows in the network. Therefore, no dedicated protocol would be needed to inform the DP about the identified flows in the PCN-domain. In any communication between ingress-nodes, egress-nodes and DP, all flows are clearly identified by their source, destination addresses and protocol identifier, if required.

Flow-rate. The number of bytes that were seen by the egress-node during one reporting period per one flow. The flow-rates are included in the report that an egress-node sends to the DP. This concerns the flows that experienced excess-marked traffic. The flow-rate for F_i is defined as (f_i, `bytes`).

Flow termination list. The flow-termination list is sent by the DP to an ingress node N_i. It contains a list of flow-identifiers per aggregate indicating the flow or flows that need to be terminated by the ingress-node. Multiple lists can be sent at once. It is defined as FTLIST((a_{i,j_1},f_1, ...,f_{l_1}), (a_{i,j_2},f_1, ...,f_{l_2}),...).

Flow-off. The flow-off message is defined as FLOWOFF without any parameters or identifiers. Note that a match must exist between the destination address to which the FLOWOFF message is sent to and the *source* part of flow-id listed in FTLIST.

Aggregate-identifiers. The identification of aggregates is based on [5], defined by the ingress-node and egress-node addresses, i.e. the 2-tuple (`i-node-address,e-node-address`).

DP identification. The identification of the DP at the ingress/egress nodes is done explicitly by defining its address to which the reports should be sent.

Combining messages. In Section 4.2, event (3), both the STATECHANGE and RATEREQUEST messages could be combined as one message.

5. Simulation of cPCN; comparison to dPCN and non-PCN network

In this section we demonstrate the usefulness and effectiveness of the proposed signaling implemented in cPCN. We developed a discrete event network simulator in C++ in which we implemented the signaling as suggested in [5]. Our goal is to have a tool available which approximates a real network to which we can add existing and future network functionality. Open source simulators like NS2, NS3 and OMNeT++ are available. However, providing these tools with new functionality would not scale and would take a considerable amount of time to add. We developed and implemented the missing signaling concerning flow termination. To be specific, we added:

1. A flow rate parameter to the feedback sent by the egress nodes to the DP, indicating the bandwidth a flow consumed during the last reporting time period.
2. A flow-termination list, sent by the DP to the ingress node to inform what flows need to be terminated, if applicable.
3. A stop-signal, sent by the ingress node to the source informing to stop the flow.

Note that since we did not implement RSVP as suggested in [1], the teardown message is not implemented to inform a source to stop a flow. The stop-message may be a RSVP teardown message in case RSVP is implemented.

We run multiple simulations in cPCN and dPCN whereby we vary the reporting time, the line delays and flows. We used common random number streams to create a set of different flows in order to compare different networks acting upon these flows. Due to the behavior of the different PCN strategies (or none in case of the non-PCN case) the outcome is different as of from some point time, since the networks decide differently (or not) on AC and FT. In Section 5.1 the simulation setup and definition of parameters is

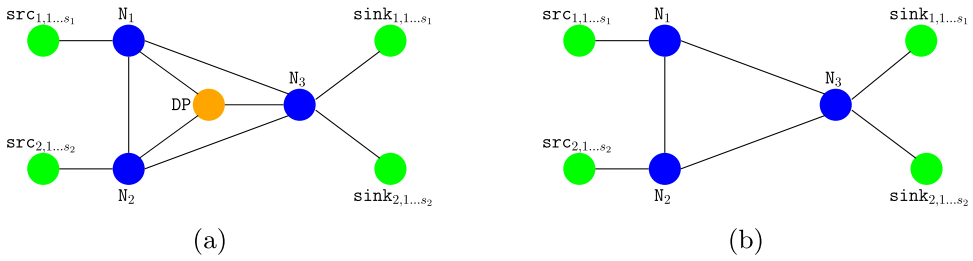

Fig. 5. PCN network with centralized (5a) and decentralized DP (5b) applied to a 3-node PCN-domain with link failure.

given and in Section 5.2 the results from all simulations are discussed.

5.1. Simulation setup and parameter choices

As stated above, the reporting time, line delay and flows will be varied. All other parameters are kept fixed (node delay, line bandwidths, simulation time, maximum number of flows, flow characteristics, link failure times, PCN thresholds, CLE limit and DSCP bits). In addition, simulations are done in the same network without PCN being active. The results from these three sets of simulations are compared and discussed. Ideally, the impact of the respective signaling architectures of cPCN and dPCN is limited and their performance differs only slightly. The simulations are primarily aimed to check this for a broad range of system parameter values. In addition we will also illustrate the benefits of the use of (c/d)PCN compared to a network without PCN. Note that our simulations are not run for the validation of PCN *itself*.

The basis of our simulations is a network consisting of three ie-nodes with sources and sinks. In cPCN (Fig. 5(a)), the DP connects to all three ie-nodes. This way all signaling between DP and ie-nodes flows exclusively through these links. In dPCN (Fig. 5(b)) all ie-nodes act as DP for their associated aggregates. Let s_1 and s_2 be the number of sources that connect to $\mathtt{N}_1$ and $\mathtt{N}_2$ respectively. One source generates one flow or multiple flows in sequence, not parallel. Each source sends traffic to one sink exclusively, i.e. source $\mathtt{src}_{i,j}$ sends traffic to $\mathtt{sink}_{i,j}$, with $i = 1, 2$ and $j = 1, \ldots, s_i$. Then the number of sinks that connect to $\mathtt{N}_3$ is $s_1 + s_2$. Details of the parameters used in our simulations are given below:

Nodes and links All node delays are set to 100 µs, the bandwidths of the links between the ie-nodes are set to 10 mbps and, if applicable, the bandwidths of the DP-node links are set to 9.999 mbps. This way the DP-node links are not chosen as best paths as a result from the Shortest Path First (SPF) algorithm. The delay on all links will be set to 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400 ms including the links between the DP and $\mathtt{N}_i$. The delay of all links from sources and sinks towards the network are fixed at 10 ms.

Flows We chose to have 160 sources, 80 connected to $\mathtt{N}_1$ and 80 connected to $\mathtt{N}_2$, i.e. $s_1 = 80$ and $s_2 = 80$. Each source produces a G.711 flow in on/off fashion. This means that no more than 7 mbps of traffic enters $\mathtt{N}_i$, see below on the PCN thresholds. The number of sources are set such, that if no extraordinary situation occurs, the amount of traffic in the network will not exceed the $\mathtt{ETM}_{\mathtt{rate}}$ per circuit, i.e. no FT occurs. If however, a link will fail, the remaining link will be over flooded and the ETM threshold will be exceeded. If the FT criterium is met, flow termination will start. A G.711 flow over Ethernet uses 87.2 kbps in bandwidth. The on/off duration is exponential distributed with means 40 and 15 s respectively. During the on-time of a flow, a stream of packets is sent at a fixed pace by the source (50 pps). During the off-time of a flow no packets are sent.

Link failure The link between $\mathtt{N}_1$ and $\mathtt{N}_3$ fails at $t = 60.22$ s and restores at $t = 100.22$ s. These numbers have been chosen such that the reporting time and the link failure/restoration do not occur at the same moment.

$\mathtt{ThM}_{\mathtt{rate}}$ and $\mathtt{ETM}_{\mathtt{rate}}$ The threshold-rate and excess-rate are set to 5.0 mbps and 9.0 mbps respectively on all inter ie-node links.

$\mathtt{CLE}_{\mathtt{lim}}$ The $\mathtt{CLE}_{\mathtt{lim}}$ is set to 0.375. Thereby, setting the admissible rate ($\mathtt{ADM}_{\mathtt{rate}}$) to 8.0 mbps. This follows from (1).

Reporting time (τ). We vary the reporting time from 50 ms to 1000 ms with 50 ms increments and from 1000 ms to 2500 ms with 500 ms increments. In order to simulate no PCN at all, the reporting time is set equal to the simulation time.

Classes of Service All PCN traffic is marked to the same class (BE). During the simulations no non-PCN traffic exists in the network.

Simulation time The simulation time is set to 150 s. This duration is sufficient to show the behavior of the signaling.

Common random numbers In order to create different flows, we varied the seed value in the pseudo-random generator of the system on which the simulations run. The same seed values are used for each set of flows per one reporting value and one line delay value.

5.2. Simulation results

The following simulations were done. In cPCN and dPCN we varied the line delay. Per line delay value we varied the reporting time and ran several simulations per reporting time. The values for the line delays and reporting times used in the simulations can be found in Section 5.1. The results of the simulations in cPCN and dPCN are given in Section 5.2.1.

In the non-PCN network, no PCN exists and therefore no reporting, no admission control and no flow termination occurs. We only varied the line delay and ran several simulations per line delay. Together with the results from the non-PCN network, the average number of goodput flows of all three architectures are discussed in Section 5.2.2.

5.2.1. Simulation results in cPCN and dPCN

Fig. 6(a)–(c) show the average number of goodput flows measured per reporting time. Goodput flows are defined as flows that travel through the network without packet loss. With larger reporting time periods, the number of goodput flows seem not to vary much.

The number of average admitted flows (Fig. 7(a)–(c)) decreases per reporting time period while the reporting time period increases, regardless of the line delays. During our simulations we found that the total time at which an aggregate is in blocking state, i.e. the blocking time, increases when the reporting time increases. This means that the 'window of opportunity' for a flow to enter the network decreases. This holds during the link failure, during

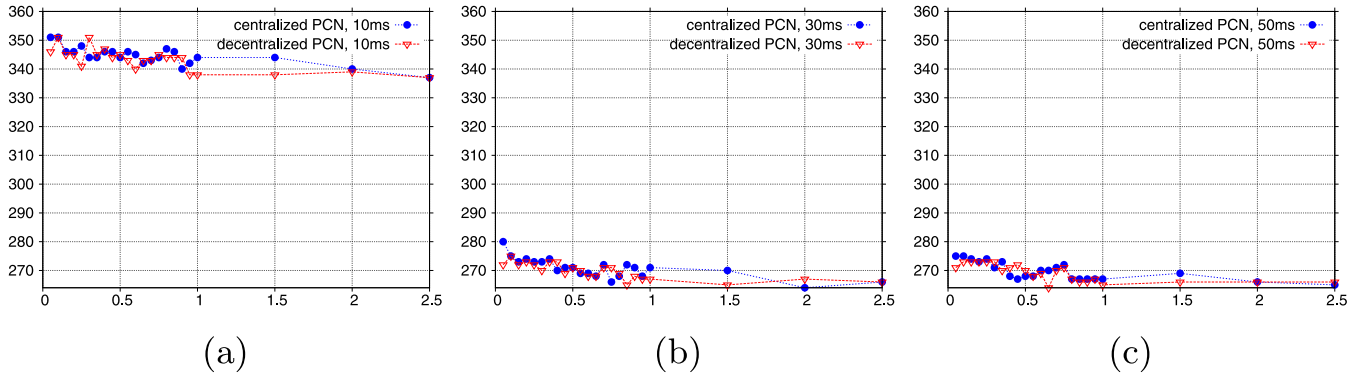

Fig. 6. Average number of goodput flows as a function of reporting time in seconds in cPCN and dPCN for line delays of 10 (6a), 30 (6b) and 50 ms (6c).

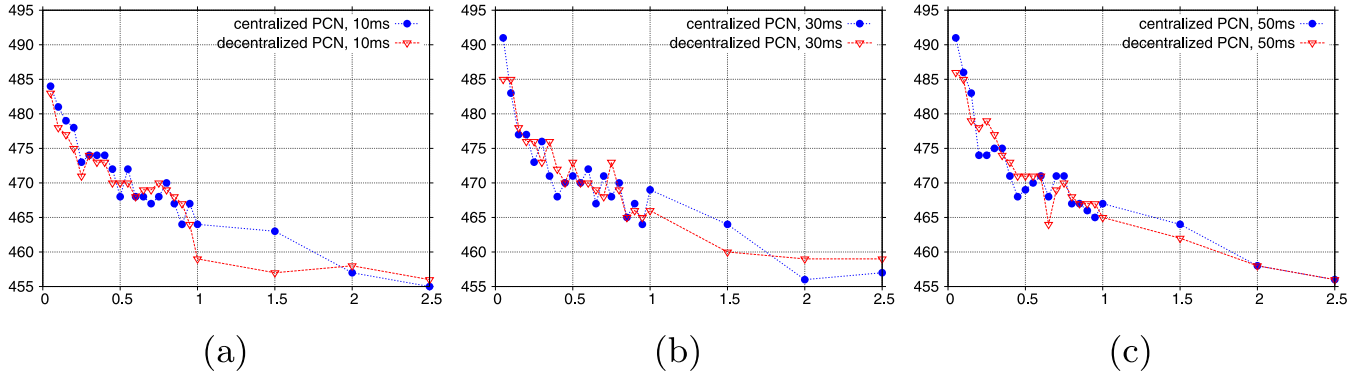

Fig. 7. Average number of admitted flows as a function of reporting time in seconds in cPCN and dPCN for line delays of 10 (7a), 30 (7b) and 50 ms (7c).

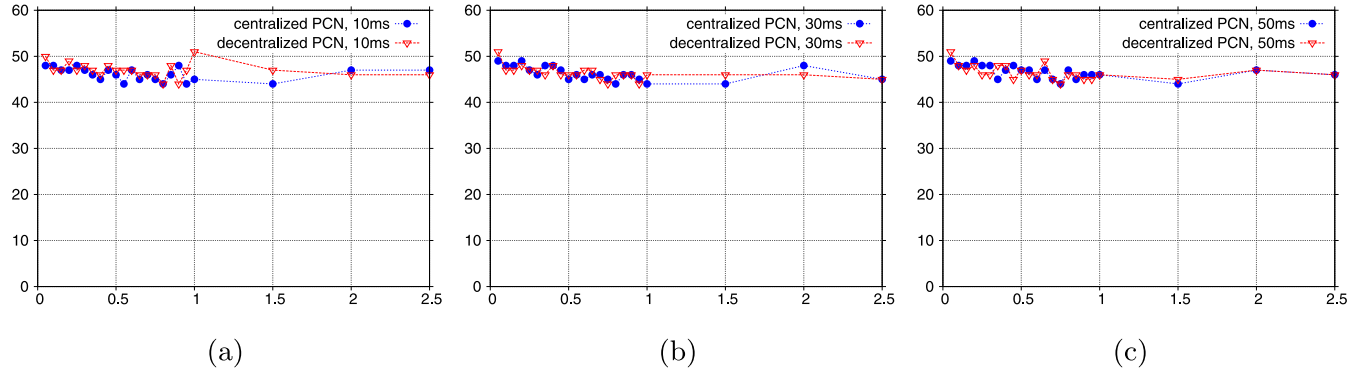

Fig. 8. Average number of blocked flows as a function of reporting time in seconds in cPCN and dPCN for line delays of 10 (8a), 30 (8b) and 50 ms (8c).

which the remaining link gets over saturated. Therefore, the network will decide to block new flows (or even terminate flows). On the other hand, with relative large reporting times and an aggregate in blocking state, more flows may end naturally unharmed. This suggests that a certain equilibrium could be reached. This is also suggested with relatively large reporting time period in the average goodput flows (Fig. 6(a)–(c)).

Fig. 8(a)–(c) show the average number of blocked flows per reporting time period. The values do not vary much (between 44 and 51), but tend to decrease with an increasing reporting time.

Fig. 9(a)–(c) show a decreasing number of terminated flows while the reporting time increases. In our simulations, FT is only applied if a link failure occurs. Since the number of blocked flows does not vary much and the average number of admitted flows decreases while the reporting time increases, the average number of terminated flows should decrease since the amount of traffic exceeding the ETM_{rate} decreases.

5.2.2. Aggregated results from cPCN, dPCN and non-PCN networks

In order to compare performances of the PCN architectures to the non-PCN network, the goodput flows of all reporting values in the cPCN and dPCN simulations are averaged per line delay. These aggregated results are given in Fig. 10. It shows the average of goodput flows summarized per line delay in the network of all simulations per architecture with the parameters mentioned in Section 5.1. Clearly, cPCN and dPCN perform very similar. We also conclude that both PCN architectures perform better than a network without PCN, since the average number of goodput flows in the PCN architectures are greater than the average number of goodput flows in the non-PCN network. With a line delay below 20 ms, the average number of goodput flows is significantly higher than seen during the line delay greater or equal than 20 ms values. Flows (packets) arrive sooner at the egress nodes with a smaller line delay. This results in sooner aggregate blocking and flow termination. Therefore, less existing flows experience packet drops.

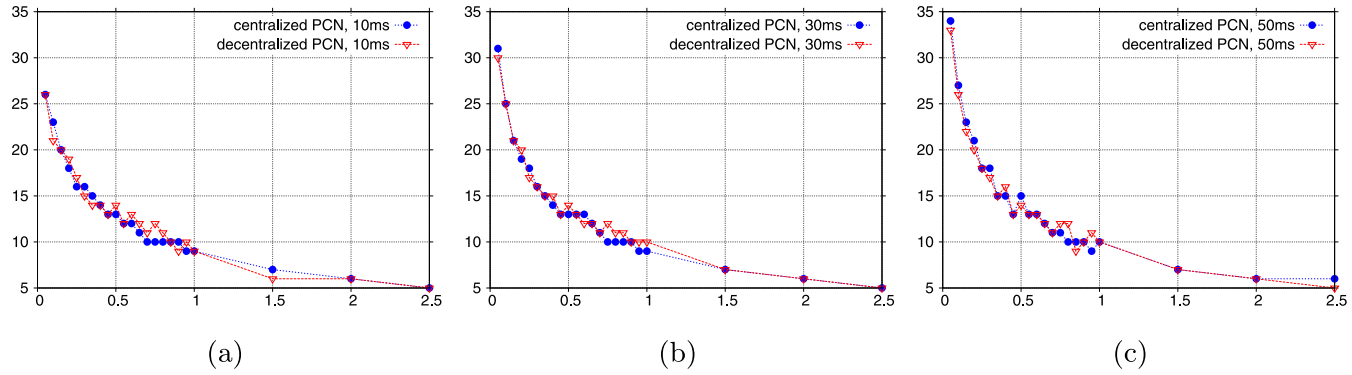

Fig. 9. Average number of terminated flows as a function of reporting time in seconds in cPCN and dPCN for line delays of 10 (9a), 30 (9b) and 50 ms (8c).

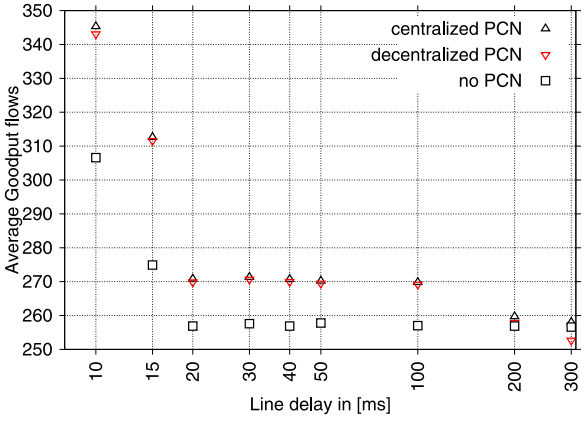

Fig. 10. Average aggregated number of goodput flows as a function of line delay in cPCN, dPCN and non-PCN architecture.

The average of goodput flows remains at the same level until a line delay of 100 ms. A further increase of the line delay leads to increasingly more damaged flows for all three cases. With even greater line delays, the performances of cPCN, dPCN and non-PCN decrease and appear to be similar.

5.3. False positives and false negatives

In general, any signaling leads to temporarily non-synchronized nodes due to delay of signaling messages. The sections below identify situations that were seen during the simulations that led to false positive and false negatives.

5.3.1. Admission control

Consider the situation whereby no link failure takes place. In cPCN, at the end of a reporting period the egress nodes send a report to the DP. If an aggregate's state should be changed, the DP informs the ingress node to change the state. This communication takes time due to line delay, serialization and processing delay at the ingress, egress nodes and the DP. While this communication takes place a new flow, arriving just after the beginning of a new reporting period but before the state change reaches the ingress node, may be admitted or blocked to an aggregate, depending on the current aggregate's state. As a consequence the number of admitted and blocked flows and therefore the number of active flows, may differ in small amounts. A false positive admission (or block) affects the load temporarily on the internal links which may lead to an increased (or decreased) amount of $\mathrm{ThM}_{\mathtt{rate}}$ or $\mathrm{ETM}_{\mathtt{rate}}$. In turn, this may lead to terminating (or not terminate) a flow if the FT criterium is met (or not met). Note that flows may disappear naturally as well. Either way, the network will protect the inelastic flows and the load will not significantly differ in the long term.

If the reporting time is too small, for example less than the round trip time between ingress and egress nodes, the local databases at the ingress, egress and DP nodes may never be synchronized. If the reporting time is too large, the protective functionality of PCN may be lost due to flows competing on bandwidth.

5.3.2. Flow termination

If a link failure occurs, assume an aggregate is in BLOCK state and flow F_1 was previously admitted and travels through this aggregate, from ingress node I to egress-node E. Flow F_1 consists of packets $P_1, \ldots, P_{n-1}, P_n$. If, at a certain point in time, I terminates F_1 upon receiving a FTLIST from the DP, I removes F_1 from its local database of admitted flows and sends a FLOWOFF message to the source of F_1. If packet P_n was sent by the source before it received a FLOWOFF message. Packet P_{n-1} arrived at I before I terminated F_1 (the delay between source and I is equal in both directions) and is considered as the last packet of F_1. So, P_{n-1} flows through the PCN-domain arriving at E. Now, P_n is interpreted as the beginning of a new flow F_2 by I and blocks it. When E sends a report to the DP. This report includes F_1 as P_{n-1} was seen at E. The DP may conclude (again) to terminate F_1, if the FT criterium is met. Since the DP does not see a difference between F_1 and F_2 (identification based on source and destination addresses), F_1 gets included in the FTLIST list which is sent to I. Here the databases of I and the DP mismatch. I does not have F_1 or F_2 in its local database of admitted flows, while the DP administers a terminated flow. Hence, the number of terminated flows differs from the number of terminated flows in dPCN. Note that the above condition may occur in dPCN.

6. Discussion, conclusions and future work

In this paper we specified the signaling in cPCN focussed on flow termination. Using extensive simulations we showed that the performance of cPCN and dPCN are similar. This means that the signaling for cPCN, defined in this paper, is effective and meaningful. Small differences were observed in the results for these architectures both in normal operation and during an extraordinary situation, which are due to different signaling and signaling delay. False positives on admission control and flow termination may happen. Despite the extra delay in the signaling, in the long run the performance of both architectures is similar as admission control and flow termination will keep the load in the network to a sustainable level. The centralized architecture leads to more complexity in the network since the information on aggregate and flow status is kept (also) at a central node. Synchronization issues may exist when signaling packets get lost. This has not been considered

in this paper. The load of the signaling is not considered in this paper. However, the load of the signaling in cPCN is expected to be less than the signaling load in dPCN due to the egress-to-any-ingress reporting in dPCN as opposed to the egress-to-DP reporting and DP-ingress signaling in cPCN.

In this paper we considered only BE marked traffic. A more granular termination of flows would be possible by considering multiple classes. Flows in lower priority classes would be terminated before terminating flows in a higher priority class. However, this paper is restricted to BE marked traffic to keep focus on the signaling. Classed-based flow termination, along with its associated signaling parameters, is considered a natural extension to the current flow termination. With multiple classes additional features come into play. Congestion management and congestion avoidance mechanisms should be considered in such case.

As mentioned before the cPCN architecture aligns well with the architecture of SDN (see e.g. [7–9]) in a sense that the control of the network is moved to a central node where all decision making is done. Bringing the (c)PCN functionalities considered in this paper into SDN will enrich SDN, but has to the best of our knowledge not been considered yet in literature. These functionalities could be implemented in a SDN-based network in several ways. For example, the cPCN DP could be added as a process to the SDN controller. It could also be added as a hardware appliance or a virtual network function communicating to the SDN controller on AC and FT. Further research is needed to investigate the details of possible implementations, including architectural implications and their performance.

References

[1] E. P. Eardley, Pre-Congestion Notification (PCN) architecture, IETF Working Group (rfc 5559) (2009).

[2] A. Charny, F. Huang, G. Karagiannis, M. Menth, T. Taylor, Pre-congestion Notification (PCN) boundary-node behavior for the controlled load (CL) mode of operation, IETF Network Working Group (rfc 6661) (2013).

[3] A. Charny, F. Huang, G. Karagiannis, M. Menth, T. Taylor, Pre-Congestion Notification (PCN) boundary-node behavior for the single marking (SM) mode of operation, IETF Network Working Group (rfc 6662) (2012).

[4] B. Briscoe, T. Moncaster, M. Menth, Encoding three Pre-Congestion Notification (PCN) states in the IP header using a single diffserv codepoint (DSCP), IETF Working Group (rfc 6660) (2012).

[5] G. Karagiannis, T. Taylor, K. Chan, M. Menth, P. Eardley, Requirements for signaling of pre-congestion information in a diffserv domain, IETF Network Task Force (rfc 6663) (2012).

[6] F. Lehrieder, M. Menth, RCFT: A termination method for simple PCN-based flow control, Springer: Network and Systems Management, 2013, doi:10.1007/s10922-013-9264-6.

[7] W. Xia, Y. Wen, C.H. Foh, D. Niyato, H. Xie, A survey on software-defined networking, IEEE Commun. Surv. Tutor. 17 (No. 1) (2015).

[8] I.F. Akyildiz, A. Lee, P. Wang, M. Luo, W. Chou, Research challenges for traffic engineering in software defined networks, IEEE Netw. (2016).

[9] M. Karakus, A. Durresi, Quality of Service (QoS) in software defined networking (SDN): A survey, Netw. Comput. Appl. 80 (2016) 200–218, doi:10.1016/j.jnca.2016.12.019.

[10] B. Briscoe, P. Eardley, D. Sonhurst, F. le Faucheur, A.C.C. Liator, J. Babiarz, K.C.C. Dudley, G. Karaiannis, A. Bader, L. Westberg, Pre-congestion notification marking - draft-briscoe-tsvwg-cl-phb-03, IETF Network Task Force (2006).

[11] J. Zhang, A. Charny, V. Liatsos, F. le Faucheur, Performance evaluation of CL-PHB admission and pre-emption algorithms - draft-zhang-pcn-performance-evaluation-01, IETF Network Task Force (2007).

[12] M. Menth, F. Lehrieder, Performance evaluation of pcn-based admission control, International Workshop on Quality of Service (IWQoS), 2008, doi:10.1109/IWQOS.2008.19.

[13] S. Latré, B.D. Vleeschauwer, W.V. de Meerssche, F.D. Turck, P. Demeester, K.D. Schepper, C. Hublet, W. Rogiest, S. Custers, W.V. Leekwijck, Design and configuration of PCN based admission control in multimedia aggregation networks, in: IEEE Global Telecommunications Conference, 2009, doi:10.1109/GLOCOM.2009.5425409.

[14] M. Menth, F. Lehrieder, Performance of PCN-based admission control under challenging conditions, IEEE/ACM Trans. Netw. (2012), doi:10.1109/TNET.2012.2189415.

[15] S. Latré, B.D. Vleeschauwer, W.V. de Meerssche, S. Perrault, F.D. Turck, P. Demeester, An autonomic PCN based admission control mechanism for video services in access networks, IEEE: Integrated Network Management-Workshops, 2009, doi:10.1109/INMW.2009.5195955.

[16] X. Zhang, Performance Evaluation of Pre-Congestion Notification, Ph.D. thesis, Cornell University, 2009.

[17] M. Menth, B. Briscoe, T. Tsou, Pre-congestion notification – new qos support for differentiated services IP networks, IEEE Commun. Mag. (2012), doi:10.1109/MCOM.2012.6163587.

[18] R. Hancock, G. Karagiannis, J. Loughney, S. van den Bosch, Next Steps In Signaling (NSIS): Framework, IETF Network Task Force (rfc 4080) (2005).

[19] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, Resource ReSerVation Protocol (RSVP), IETF Network Task Force (rfc 2205) (1997).

[20] G. Karagiannis, A. Bhargava, Extensions to generic aggregate RSVP for IPv4 and IPv6 reservations over Pre-Congestion Notification (PCN) domains, IETF Network Working Group (rfc 7417) (2014).

[21] K. Chan, J. Seligson, D. Durham, K. McCloghrie, F. Reichmeyer, R. Yavatkar, A. Smith, COPS Usage for policy provisioning (COPS-PR), IETF Network Task Force (rfc 3084) (2001).

[22] V. Fajardo, J. Arkko, J. Loughney, G. Zorn, Diameter based protocol, Internet Engineering Task Force (rfc 6733) (2012).

Frank Wetzels has a B.S. degree in Computer Technology and a M.S. degree in Mathematics. He works as a network engineer for over 20 years and holds several industrial recognized certifications like CCIE and CEH. His expertise in networking engineering lies in the field of security and routing and switching focused on service provider networks. Currently, he is a Ph.D. candidate at the VU University of Amsterdam. His research interests include performance analysis and modeling of networking technology.

Hans van de Berg (M.Sc. and Ph.D. degree in Applied Mathematics from the University of Utrecht, The Netherlands, in 1986 and 1990 respectively) has more than 25 years of experience in ICT research and innovation. His main contributions are in the field of design and performance optimization of communication networks and service platforms, with special emphasis on autonomous control methods since the last 10 years. He has been active in many national and European research projects and platforms (FP3-FP7, COST, ITEA) and is co-founder and vice-chair of the currently running COST Action IC1304 Autonomous Control for a Reliable Internet of Services (COST ACROSS). He has published more than 150 refereed papers in international journals and conference proceedings. From 1990 until 2002 Hans van den Berg was with KPN Research. Since then he is with TNO (Dept. Performance of Networks & Systems) and holds a part-time position as full professor in Computer Science at the University of Twente. Since 2016 he is also affiliated with CWI, the Dutch national research institute for mathematics and computer science.

Joost Bosman (1983) received his M.Sc. (cum laude) degree in Business Mathematics and Informatics in 2009 from the VU University Amsterdam. After he obtained his Ph.D. degree at VU university in 2014 he started as a post-doc researcher at Centrum Wiskunde & Informatica. His research interests include performance and QoS modeling of ICT systems, and queueing theory. In 2010 he co-organized the 72nd European Study Group Mathematics with Industry in Amsterdam.

Rob van der Mei is a full professor at the VU University Amsterdam, the head of the research theme Logistics and the Industrial Liaison Officer at CWI. Before going to academia, he has been working for over a decade as a consultant and researcher in the ICT industry, working for PTT, KPN, AT&T Bell Labs and TNO ICT. He is the initiator and leader of the project From Reactive to proactive Planning of Ambulance Services (REPRO), and a co-founder of the spin-off company Stokhos Emergency Mathematics. His research interests include performance modeling and scalability analysis of ICT systems, logistics, grid computing, revenue management, military operations research, sensor networks, call centers, queueing theory and applications of BigData. He is the co-author of some 150 papers in journals and refereed proceedings.