DCP-EW: Distributed Congestion-control Protocol for Encrypted Wireless Networks

Xiaolong Li Homayoun Yousefi'zadeh
Department of EECS
University of California, Irvine
[xiaolonl, hyousefi]@uci.edu

***Abstract*— VCP suffers from a relatively low speed of convergence and exhibits biased fairness in moderate bandwidth high delay networks due to utilizing an insufficient amount of congestion feedback. Our previous work Double-Packet Congestion-control Protocol (DPCP) addressed this problem by increasing the amount of the feedback distributed over two ECN bits in the IP header of a pair of packets. However, DPCP faces deployment obstacles in encrypted wireless networks due to the fact that it relies on partial information in the TCP header and the TCP header information is lost when crossing encryption boundaries. Furthermore, wireless networks are characterized by both error- and congestion-caused loss. Our previous work has revealed that the efficiency of DPCP, and for that matter any congestion control protocol, over wireless networks may be reduced as the result of not being able to differentiate between two types of loss. In this paper, we propose an alternative congestion control protocol to which we refer as Distributed Congestion-control Protocol for Encrypted Wireless (DCP-EW) networks. DCP-EW is capable of efficiently operating in encrypted wireless networks while preserving all of the benefits of DPCP for wired networks. It does so by passively utilizing the IP Identification field of a packet header instead of the TCP header in conjunction with a heuristic algorithm to differentiate between different sources of loss. We implement DCP-EW in NS-2 and the Linux Kernel. We demonstrate the performance improvements of DCP-EW compared to DPCP and VCP through simulation and experimental studies.**

***Index Terms*— Congestion Control, Encrypted Wireless Networks, ECN, High BDP, VCP, DPCP.**

I. Introduction

It has been shown that conventional TCP and end-to-end TCP-based Active Queue Management (AQM) schemes perform poorly in high Bandwidth-Delay Product (BDP) networks [1]. Over the past few years, an abundant of techniques has been developed to improve the efficiency and fairness of TCP. Examples include the works of [2], [3], [4] using algorithms to adaptively adjust the sending window size, and [5], [6], [7], [8] employing alternative congestion signals. However, due to their integrated controller design, these techniques often fail to achieve both efficiency and fairness [1].

By decoupling efficiency control from fairness, eXplicit Congestion-control Protocol (XCP) [9] and Variable-structure Congestion-control Protocol (VCP) [10] can achieve high utilization, low persistent queue length, insignificant packet loss rate, and sound fairness depending on the heterogeneity characteristics of a network. While XCP requires the use of a large number of IP packet header bits to relay congestion information thereby introducing significant deployment obstacles, VCP only uses the two existing ECN bits in the IP header to encapsulate three congestion levels. Given that VCP demands the use of no extra bits in the IP header, it represents a more practical alternative of deployment than XCP.

However, VCP can only deliver limited feedback to end hosts since two bits can at most represent four levels of congestion. In order to avoid sudden bursts, VCP has to control the growth of transmission rates by setting artificial bounds. The latter, yields slow convergence speeds and high transition times. Moreover, due to the use of fixed parameters for fairness control, VCP exhibits poor fairness characteristics in high delay networks.

Most recently, several works have attempted at addressing the problem associated with VCP limitations by increasing the amount of feedback. While the work in MLCP [11] proposes using 3 bits to represent the Load Factor (LF), the UNO framework [12] proposes another alternative to increase the amount of feedback by passively utilizing information in IP Identification (IPID) field. In contrast, our previous work DPCP [13] proposes a distributed framework that allows for using no more than 2 ECN-bits to deliver a 4-bit representation of the LF. That said, DPCP needs to access partial information in the TCP header in order to be able to efficiently distribute and reassemble the LF. However, in encrypted networks protected by IPSec, TCP header information is lost when crossing encryption boundaries. Thus, DPCP cannot operate in such encrypted networks. Furthermore, wireless networks are characterized by fading related error-caused loss in addition to queuing related congestion-caused loss. Experiments have shown that the performance of any congestion control protocols relies on appropriate reaction to loss according to its source. Like VCP, DPCP reacts to loss without differentiating between the sources of loss and thus performs inefficiently over wireless networks.

In this paper, we propose a new congestion control protocol that improves the design of DPCP. In contrast to DPCP, our new protocol to which we refer as Distributed Congestion-control Protocol for Encrypted Wireless (DCP-EW) networks proposes two new schemes: i) a novel distributed scheme that allows for operation within encrypted networks, and ii) a new heuristic loss differentiating scheme that can distinguish between error-caused loss and congestion-caused loss. Notably, these new schemes are added to DCP-EW while preserving all of the benefits of DPCP. More specifically, DCP-EW can still provide the sender with fine grain congestion levels using no more than the two ECN bits in the IP header. In DCP-EW, a congestion level is carried by a chain of two packets and each packet provides two bits out of four bits of information associated with a con-

This work was sponsored by a grant from the Boeing Company.

gestion level. Utilizing a distributed scheme that deviates from that of DPCP, routers compute and distribute a congestion signal into two packets. A congestion level can be specified by concatenating a group of two ECN bits together from a pair of packets at an end node. Incorporated with a novel heuristic algorithm, DCP-EW can appropriately react to congestion-caused loss while avoiding unnecessary reductions of the sending window sizes in response to error-caused loss.

Finally, we implement DCP-EW in both NS-2 [14] and the Linux Kernel. Through both simulation and experimental studies, we demonstrate that DCP-EW is able to achieve a performance comparable to that of DPCP in wired networks. We also demonstrate that DCP-EW can operate in IPSec encrypted networks while significantly outperforming DPCP in terms of convergence speed and fairness in wireless networks characterized by tandem loss.

The rest of the paper is organized as follows. In Section II, we review the design methodology of VCP and DPCP along with their shortcomings in order to describe the motivation for the design of DCP-EW. In Section III, we present the two novel components of DCP-EW compared to DPCP. Experimental studies are presented in Section IV. In Section V, we review the related work to DCP-EW. Finally, we present several conclusions in Section VI.

II. Fundamentals of VCP and DPCP

Rather than using an integrated controller like TCP, VCP attempts at decoupling efficiency and fairness aspects of congestion control and operating in three congestion regions. While VCP retains the sliding window and acknowledgment (ACK) mechanisms of TCP, its window management mechanism is quite different than that of TCP. Instead of using the slow start and congestion avoidance algorithm of TCP, VCP regulates the value of $cwnd$ with different congestion control policies defined according to the level of congestion in the network. VCP represents the network congestion status by a load factor which is further mapped into one of three congestion levels labeled as low-load, high-load, and overload. The design of VCP allows for encoding the value of the LF into two ECN bits in the IP packet header. The LF is computed and mapped into one of the three congestion levels mentioned above at a VCP router. Once a data packet arrives, the VCP router extracts the congestion level associated with its most congested upstream link from the ECN bits of the packet itself. It then updates the ECN bits of the packet only if its downstream link is more congested than what is already indicated by the ECN bits of the packet. Eventually, the data packet will carry the congestion level of the most congested link of its session. At the receiver, the congestion level is retrieved and sent back to the sender via an ACK packet. Consequently, VCP applies three congestion control policies: Multiplicative Increase (MI) in the low-load region, Additive Increase (AI) in the high-load region, and Multiplicative Decrease (MD) in the overload region. While the MI operation is utilized to eliminate TCP's slow start behavior, the AI and MD operations attempt at preserving the fairness characteristic of TCP. Since VCP can only provide limited feedback to the sender, its efficiency and fairness characteristics are negatively impacted in moderate bandwidth high delay network operation scenarios.

Unlike VCP, DPCP uses four bits to represent the LF. Although DPCP increases the amount of feedback, it utilizes the two ECN bits of a pair of packets in order to encode the LF in a distributed way. For a given LF, the packet that carries the Most Significant Bits (MSBs) of the LF is referred to as MSP. Similarly, the packet that carries the Least Significant Bits (LSBs) of the LF is referred to as LSP. Each packet has a sequence (seq) number and an acknowledge (ack) number in its TCP header. During transmission, these two numbers never change. Thus the relative order of these two numbers can be used as a binary indication to tell if a packet is MSP or LSP. More specifically, if the seq number has a greater value than the ack number, then the packet is the MSP. Otherwise the packet is the LSP. Furthermore, DPCP maintains an MSP flag at the end nodes. The end nodes flip over the MSP flag of every packet to indicate if the next packet should be MSP/LSP. Based on the value of MSP, end nodes may swap the value of seq and that of ack in order to use the packet as MSP or LSP and thus yield an interleaved packet flow with the pattern "MSP:LSP:MSP:LSP:...". Once a packet arrives at a router, the router identifies a packet as MSP or LSP by checking the relative order of the seq and ack of the packet. Then, the router assigns either MSB or LSB bits of the associated LF to the packet depending on whether it is MSP or LSP. This way, DPCP can provide a more accurate feedback to the sender.

III. DCP-EW: Distributed Congestion Control Protocol for Encrypted Wireless Networks

As presented earlier, the design of DCP-EW is motivated by two observations. First, most feedback based congestion control protocols either require the use of multiple bits in the IP header or even access to headers of the protocols above the IP layer, thereby facing deployment challenges in encrypted networks. Second, most congestion control protocols are designed for wired networks and treat both types of loss as congestion-caused loss. While error-caused losses are typically absent in wired networks, they are common in wireless networks. Experiments show that reacting to error-caused and congestion-caused loss, can significantly decrease the performance of any congestion control protocol. Thus, the target operating environments of DCP-EW are IPSec-based encrypted wireless networks. The latter means that only eight bits of the IP header, two ECN bits and six Type of Service (ToS) bits, can bypass the encryption boundaries and are available for end to end signaling. As the ToS bits are reserved for signaling differentiated services as oppose to congestion control, DCP-EW will only use the two ECN bits of the IP packet header for carrying congestion control signaling feedback.

A. Overview

Relying on two new schemes, DCP-EW extends DPCP to work efficiently in encrypted wireless networks. First and albeit the fact that DCP-EW uses a double packet four bit representation of the LF just like DPCP does, it introduces a packet ordering management strategy that is quite distinct from that

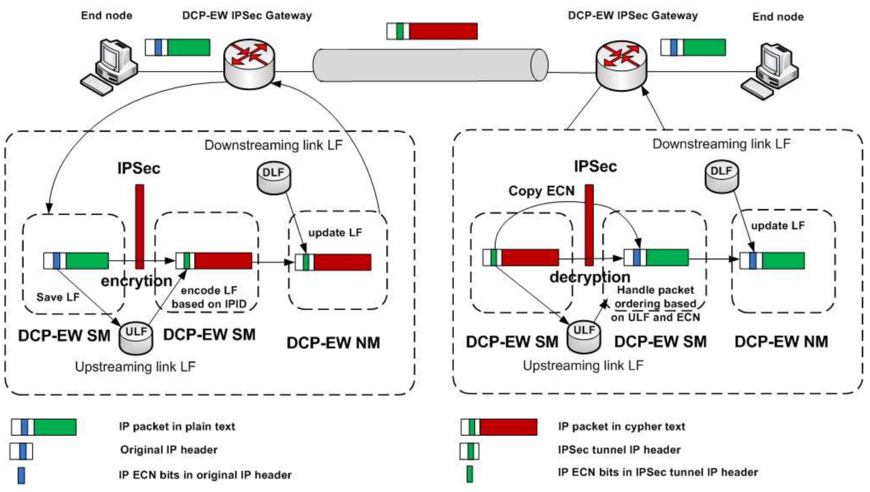

Fig. 1. An example scenario of using DCP-EW over an IPSec tunnel.

of DPCP. Unlike DPCP, DCP-EW does not rely on the TCP header to manage packet ordering. Instead, it only utilizes the information available in the IP header and only manipulates two existing ECN bits to carry congestion information. The detail of new packet ordering management scheme of DCP-EW will be presented in the next subsection. Second, DCP-EW utilizes a heuristic scheme for differentiating error-caused loss from congestion-caused loss. This heuristic scheme runs at the transmitting side and maintains the history information of congestion status over the bottleneck link of a path. Upon detection of loss, the heuristic scheme makes an identification of the source of loss based on the saved history information. Other components of DCP-EW such as the definitions of congestion levels, handling exceptions as well as encoding and decoding scheme remain the same as those of DPCP. We refer the interested reader to [13] for the description of common components. In the following two subsections, we present the novel aspects of DCP-EW, namely, how the protocol manages packet ordering and how it differentiates between two types of loss.

B. Packet Ordering Management

As DCP-EW distributes the LF into two packets, a binary signal is enough to determine packet ordering. However, no free bit is available in the IP header for such signaling. That said, the IPID field of the IP header originating from a host is either monotonically increasing or chosen uniformly at random. In either case, the LSB of IPID flips over quickly enough to be used for signaling MSP/LSP. Specifically, DCP-EW only uses the LSB of the IPID field. Further, the use of IPID field bits is passive, i.e., the bit values are inspected but not changed by DCP-EW. In contrast to DPCP, DCP-EW uses the LSB of IPID field in order to differentiate MSP from LSP at the receiving end, instead of swapping TCP seq and ack numbers. Namely, a packet with an LSB value of zero is used as the MSP and a packet with an LSB value of one is used as the LSP. As mentioned above, the value of the IPID is set by the IP protocol either incrementally or according to a uniform random distribution. In the former case, the LSB bit flips over for any pair of consecutive packets which is perfect for differentiating MSP from LSP. In the latter case and despite the fact that the LSB bit might not flip over in every pair of consecutive packets, DCP-EW uses the first packet with an LSB value of zero for carrying MSP and the first packet with an LSB value of one for carrying LSP. As evidenced in our experiments, it is safe to assume that bit flips, with a probability of 0.5, occur quick enough with respect to necessary congestion reaction speed specially over large BDP networks. In what follows, we explain how DCP-EW operates in IPSec encrypted networks. Assuming that at the encrypted boundaries, only two ECN bits can pass the boundary.

C. Operation with IPSec

IPSec operates in two modes: transport mode and tunnel mode. In the transport mode, the original IP header is kept after getting authenticated by IPSec. Thus, DCP-EW can still access IPID and ECN bits as usual in IPSec transport mode. In contrast, the entire packet is encrypted and authenticated in IPSec tunnel mode. As a result, the original IP header becomes invisible in the encrypted packet. Since the LSB bit of the IPID in the original IP header may not necessarily be the same as the one in the new IP header, DCP-EW utilizes the IPID only on the Cypher Text (CT) side but not on the Plain Text (PT) side for packet ordering. In what follows, we present the details of the operation of DCP-EW in IPSec tunnel mode. As DCP-EW will be installed and configured at the IPSec router, it is safe to assume that DCP-EW will have access to both CT and PT headers of a packet. Furthermore, because the operations of DCP-EW in the PT side are similar to that presented in [15], we only focus on the operation of the IPSec router over encryption boundaries and the IPSec tunnel. Specifically, DCP-EW provides two router modules: i) Security Module (SM) running only on IPSec routers that cooperates with IPSec gateways, and ii) Normal Module (NM) running on both IPSec gateways and other routers. Fig. 1 illustrates a scenario of using DCP-EW over an IPSec tunnel.

Assuming an FTP or a comparable connection has been established, the flow of events at the IPSec gateways is as follows: 1) A DCP-EW packet arrives at the ingress of an IPSec gateway. Before the packet goes to the IPSec module for encryption, DCP-EW SM will first catch the packet, save the packet ordering information, i.e., MSP/LSP and the value of the LF as indicated in the ECN bits. Then DCP-EW SM delivers the packet to the IPSec module. After the new IP header is generated and ready to be transmitted through the tunnel, DCP-EW SM catches the outgoing packet again and encodes ECN bits with MSB/LSB bits of the saved LF depending on the LSB bit of the IPID in the new IP header. Note that, after the original IP header is encrypted, DCP-EW has no idea of if the new packet is a TCP packet or a packet using another protocol, e.g., UDP. Thus, DCP-EW encodes ECN bits regardless of the original protocol type, which introduces overhead for non-TCP packets. In fact, this is the tradeoff between efficiency and protocol complexity. That said, we note that the resulting overhead is not significant because i) it is only introduced when transmitting over IPSec tunnels; and ii) it is only associated with the operations of encoding an LF.

2) At the output interface of the ingress IPSec gateway, DCP-EW NM takes over. DCP-EW NM compares the LF in the packet with the LF of its downstream link interface and updates the LF in the packet if necessary following the algorithm introduced in [15].

3) At the intermediate router on the CT side, DCP-EW NM operates as DPCP router module except that DCP-EW uses the LSB bit of IPID to identify MSP/LSP.

4) At the egress of the IPSec gateway and before the encrypted packet goes to the IPSec module for decryption, DCP-EW SM will catch the packet and save the LF value as indicated by the ECN bits of the packet. Note that after the packet is decrypted, the IPSec module will copy the ECN bits from the new IP header to the original IP header on the PT side. However, the packet ordering information cannot be simply transferred to the PT side. While DCP-EW SM can access both CT and PT side, DCP-EW SM dedicates to change the contents of the packet as minimally as possible. Simply put, DCP-EW SM does not directly pass any bits from the CT side to the PT side. Note that, the LSB bit of the IPID in the original IP header is not necessarily the same as the one in the new IP header. Thus, instead of changing the value of the LSB bit of the IPID field in the original IP header for the purpose of matching the one in the IP header used by the IPSec tunnel, DCP-EW uses the relative order of the TCP seq and ack numbers as the indication of MSP/LSP after the original IP header is retrieved. In this way, DCP-EW will not change any bits in the IP header of the decrypted packet. Furthermore, DCP-EW SM has to keep a copy of the LF of the upstream link of the egress IPSec gateway for each IPSec tunnel. DCP-EW SM inspects the ECN bits in the packet and compares it with the MSP/LSP of the saved copy of the LF of its upstream link. Based on the results of the comparison, DCP-EW SM manipulates the seq and ack numbers in order to mark the packet as MSP or LSP. Then the packet is delivered to DCP-EW NM. DCP-EW NM updates the ECN bits according to the LF of its downstream link following the operating mechanism of DPCP.

D. Loss Differentiation Heuristic Algorithm

Intuitively, a sender can build knowledge about whether the network is congested as it keeps receiving feedback from its intended receiver. Given the fact that the feedback is updated with the receipt of every ACK, it is reasonable to assume that the congestion status of a network can be continuously tracked by the sender. It is specially important to realize that a congestion-caused loss event has a much longer duration than an error-caused loss event. Relying on the above fact, the heuristic algorithm of DCP-EW assumes that a sender can identify the cause of a loss by keeping track of the status of the network. In order to track the status of the network, the heuristic algorithm proposes maintaining a revolving congestion history Bit Map (BM) of size N at the sending side. Upon the receipt of an ACK, the bit at position $BM(1)$ is dropped, the bit at position $BM(i)$ with $i \in \{1, \cdots, N\}$ is shifted to the left so it takes the position of bit $BM(i-1)$, and the bit at position $BM(N)$ is set to 1 if the new ACK indicates congestion or otherwise to 0. If at any time, the right most T consecutive bits with $T \leq N$ are set to 1 in the bit map, a binary flag called Congestion Flag (CF) is set to 1. Otherwise, the flag is set to 0. Upon detection of a loss, if CF flag is set, then the loss is safely determined as a congestion-caused loss triggering an MD operation to $cwnd$. Otherwise, the loss is considered to be an error-caused loss and the sender simply maintains the current $cwnd$. In the case of DCP-EW, the link LF is encapsulated in ACK packets and the $OVER\_LOAD$ represents a LF beyond 100%. Thus, $OVER\_LOAD$ is used as the indicator of congestion. According to our experiments, setting N to 32 and T to 16 represent optimal choices. We note that with our choices of values, maintaining the revolving history bit map only requires 4 bytes of storage on a per router basis. Fig. 2 illustrates the operation of the heuristic algorithm of DCP-EW.

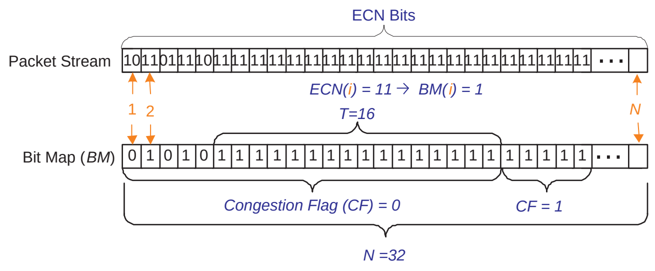

Fig. 2. An illustration of the loss differentiation heuristic algorithm of DCP-EW.

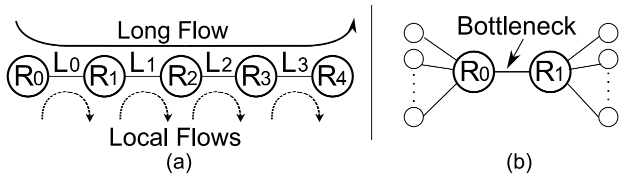

Fig. 3. An illustration of (a) parking lot and (b) dumbbell topologies used in our experiments.

IV. Performance Evaluation

In this section, simulation studies and experimental studies of DCP-EW are presented. We implement DCP-EW in both NS-2 simulator and Linux Kernel. Performance of DCP-EW, DPCP, and VCP are compared in terms of efficiency and fairness. Since DCP-EW is proposed as an extension of DPCP for

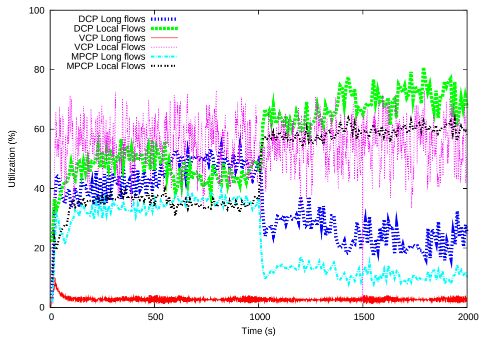

Fig. 4. A performance comparison of DCP-EW, DPCP, and VCP over link #0.

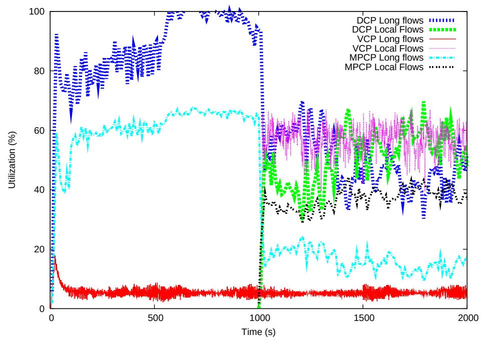

Fig. 5. A performance comparison of DCP-EW, DPCP, and VCP over link #2.

encrypted wireless networks, our target environment is characterized by moderate bandwidth ($2 - 10Mbps$) high delay ($200 - 1000ms$) lossy links. The wireless effects are introduced by utilizing the temporally correlated Gilbert Elliott (GE) model presented in our previous work [16].

A. Simulation Studies

In this subsection, we compare the performance of DPCP and VCP over a four bottleneck parking lot topology as illustrated by Figure 3(a). All of the links have a one-way delay of $250ms$ and a bandwidth of $4Mbps$ except L_2 that has a bandwidth of $2Mbps$. The GE model is applied on a per link basis in order to introduce an average loss rate 5% [1] for each link. There are two types of aggregate FTP flows traversing the topology. The first type is referred to as a Long Flow and represents the combined traffic of 30 FTP flows traversing all of the links in the forward left-to-right direction. The second type is referred as to as a Local Flow. There are four Local Flows each of which representing 10 FTP flows traversing each individual link in the forward direction. Except those flows that traverse link L_2 and start after 1000 seconds, all other Local Flows start at the beginning of the experiments.

Note that if no wireless loss is introduced, DCP-EW and DPCP achieve nearly identical performance as they share same control policy. With the heuristic scheme, DCP-EW can significantly improve the performance of DPCP over a lossy link.

[1]Note that to increase the visibility of the figure, a relatively low loss rate is introduced in this subsection.

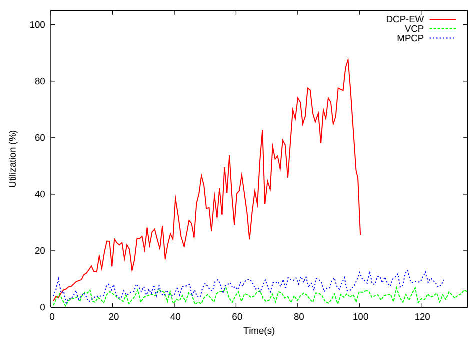

Fig. 6. A performance comparison of DCP-EW, DPCP, and VCP over the bottleneck link of our experimental dumbbell topology.

Fig. 4 and Fig. 5 show the bandwidth split ratio of VCP, DPCP, and DCP-EW respectively.

Ideally, during the first 1000 seconds, both Long and Local Flows are to equally split the bandwidth of a shared link. Starting from 1000-th second when an extra Local Flow starts at link #2, the utilization of Long Flows at Link #0 should drop to 25% while the utilization of Local Flows should go up to 75%.

In Fig.4, VCP exhibits a biased fairness characteristic splitting the bandwidth of link #0 with a ratio of 15 to 1. While DPCP demonstrates a significantly better fairness characteristic than VCP, it shows inefficiency in terms of the bandwidth utilization due to the effect of its reaction to loss. In contrast, DCP-EW shows both good fairness and efficiency.

At link #2, we expect to see a near 100% bandwidth utilization for Long Flows during the first 1000 second and a split of 50% in the last 1000 seconds between Long and Local Flow when the Local Flow joins. As illustrated by Fig. 5, both DCP-EW and DPCP show good fairness and responsiveness, although DCP-EW outperforms DPCP in terms of bandwidth utilization. To the contrary, the bandwidth split ratio does not change even when Local Flows are turned on in the case of VCP showing that VCP fails to achieve fairness in high BDP multiple bottleneck topologies serving flows with heterogeneous RTTs.

B. Experimental Studies

In this subsection, we describe our implementation of DCP-EW in the Linux Kernel. The implementation approach follows that of VCP as described in [15]. Again, we introduce packet loss using our GE error model implementation in the Linux Kernel. In this section, we do present our experimental study conducted over a real testbed comparing the performance of VCP, DPCP, and DCP-EW. Due the limitation of space, we only present the results associated with a single bottleneck scenario. We use a dumbbell topology (Fig. 3 b) with the settings used for experiments matching those of [15]. While not shown here, the performance of DCP-EW in multi-bottleneck scenarios follows the pattern shown in our simulation studies.

Fig. 6 compares the bandwidth utilization of VCP, DPCP, and DCP-EW over the single bottleneck link. In our experiments, a loss rate of up to 30% is introduced. Thus, both DPCP and VCP fail to open the $cwnd$ efficiently in the absence of the heuristic scheme, and therefore exhibit a low utilization characteristic. Note that while DPCP achieves a higher bandwidth utilization than VCP, it demonstrates oscillations due to its inappropriate reaction to error-caused loss. The improvement comes from the faster recovery speed of DPCP in contrast to VCP. In contrast, DCP-EW can identify the source of a loss and ignore error-caused loss. In the figure, DCP-EW can achieve a significantly better bandwidth utilization than both DPCP and VCP although it shows oscillations due to the associated retransmissions and timeouts.

V. Related Work

In this section, we review some of the literature work most closely related to DCP-EW. Since VCP was proposed, it has received a significant attention due to its deployment potential. The works of [16], [15] evaluated the performance of VCP in wireless networks and highlighted several limitations of VCP. Our recent work of [13] proposes a distributed approach that can overcome the limitations of VCP by increasing the amount of feedback to the sender. By distributing a 4-bit representation of the LF into two consecutive packets, DPCP only needs to use two ECN bits in one packet preserving the deployment potential of VCP. However, DPCP requires access to TCP header in order to perform encoding and decoding of the LF. The latter introduces difficulties for working in encrypted networks. It is also important to note that all VCP alternatives are faced with similar deployment issues in encrypted networks. In contrast, DCP-EW proposed in this paper is capable of working in encrypted networks by using an alternative packet ordering management scheme. As DCP-EW also distributes the LF into two consecutive packets the same way as DPCP does, this work can be viewed as an extension of DPCP for wireless networks.

Besides DPCP from which DCP-EW is derived, the closest bodies of work in congestion control to DCP-EW include MLCP [11] and UNO [12]. The MLCP [11] analyzed the control algorithm of VCP and proposed a multi-level load-factor based protocol to increase the feedback information of VCP. However, MLCP requires the use of extra bits in the IP header. The UNO framework [12] utilizes the IPID field to passively encode the LF. The passive nature comes from a fact that the UNO framework does not modify the value of the IPID field. In DCP-EW, the idea of passively using the LSB bit of the IPID field is inspired by the UNO framework. Nonetheless, while the work of UNO may seem to share a similar idea with DCP-EW, it differs from DCP-EW in several aspects. First, although UNO passively utilizes existing bits in the IPID field of the IP header, it introduces deployment issues. For example, UNO will not work in certain encrypted networks where only 6 ToS and 2 ECN packet header bits can pass through encryption boundaries. In contrast, DCP-EW only requires the use of two ECN bits in each packet. Second, UNO senders need to collect at least 8 specific packets translating to an average of $8 \ln 8 = 24$ consecutively transmitted packets in order to derive the maximum congestion level before regulating $cwnd$, while DCP-EW senders perform regulations on a per-ACK basis. Over lossy wireless links, consecutive loss of packets associated with the maximum LF yield an oscillatory behavior in the case of UNO. Most importantly, DCP-EW provides a loss identification algorithm to enable proper reaction to loss depending on its cause, while other VCP alternatives have no such capability.

VI. Conclusion

In this paper, we proposed DCP-EW as an extension of DPCP. We demonstrated how DCP-EW overcomes the limitations of DPCP by using an alternative packet ordering management scheme. Rather than accessing the TCP header, DCP-EW passively inspected the LSB bit of the IPID field in the IP packet header to identify whether a packet is the MSP or LSP in a packet pair sequence. Furthermore, DCP-EW utilized a heuristic loss identification scheme to differentiate error-caused loss from congestion-caused loss such that it can appropriately react to loss. We implemented DCP-EW in both NS-2 and the Linux Kernel. Through both simulation and experimental studies, we demonstrated that the fairness and efficiency characteristics of DCP-EW are comparable to those DPCP in wired networks. We also demonstrated that in high BDP networks, both DCP-EW and DPCP significantly outperform VCP in terms of fairness and efficiency. As the main differentiating factors, we showed that i) unlike DPCP, DCP-EW can operate over IPSec encrypted networks, and ii) relying on its heuristic loss identification algorithm, DCP-EW can significantly outperform DPCP in wireless environments characterized by tandem loss.

References

[1] M. Goutelle, Y. Gu, and E. He, “A Survey of Transport Protocols other than Standard TCP,” 2004, https://forge.gridforum.org/forum/forum.php?forum id=410.

[2] L. Xu, K. Harfoush, and I. Rhee, “Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks,” in *Proc. of the IEEE INFOCOM*, 2004.

[3] I. Rhee and L. Xu, “CUBIC: A New TCP-Friendly High-Speed TCP Variant,” in *Proc. of the PFLDNet’05*, Feb. 2005.

[4] S. Floyd, “HighSpeed TCP for Large Congestion Windows,” Aug. 2002.

[5] D. Leith and R. Shorten, “H-TCP: TCP for High-speed and Long-distance Networks,” in *Proc. of the PFLDNet’04*, Feb. 2004.

[6] T. Kelly, “Scalable TCP: Improving Performance in HighSpeed Wide Area Networks,” Feb. 2003, available at http://wwwlce.eng.cam.ac.uk/ctk21/scalable/.

[7] C. Jin, D. Wei, and S. Low, “FAST TCP: Motivation, Architecture, Algorithms, Performance,” in *Proc. of IEEE INFOCOM*, 2004.

[8] S. Bhandarkar, S. Jain, and A. Reddy, “Improving TCP Performance in High Bandwidth High RTT Links Using Layered Congestion Control,” in *Proc. of the PFLDNet’05*, Feb. 2005.

[9] D. Katabi, M. Handley, and C. Rohrs, “Congestion Control for High Bandwidth-Delay Product Networks,” in *Proc. ACM SIGCOMM*, Aug. 2002.

[10] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, “One More Bit Is Enough,” in *Proc. ACM SIGCOMM, 2005*, Aug. 2005.

[11] I. A. Qazi and T. Znati, “On the design of load factor based congestion control protocols for next-generation networks,” in *Proc. of the IEEE INFOCOM 2008*, Apr. 2008.

[12] N. Vasic, S. Kuntimaddi, and D. Kostic, “One Bit Is Enough: a Framework for Deploying Explicit Feedback Congestion Control Protocols,” in *Proc. of the First International Conference on COMmunication Systems and NETworkS (COMSNETS)*, Jan. 2009.

[13] X. Li and H. Yousefi’zadeh, “Distributed ECN-Based Congestion Control,” in *Proc. of the IEEE ICC 2009*, June 2009.

[14] -, “UCB/LBNL/VINT Network Simulator - ns (version 2),” available at www.mash.cs.berkeley.edu/ns/.

[15] X. Li and H. Yousefi’zadeh, “An Implementation and Experimental Study of the Vraiable-Structure Congestion Control Protocol (VCP),” in *Proc. of the IEEE MILCOM, 2007*, Oct. 2007.

[16] H. Yousefi’zadeh, X. Li, and A. Habibi, “An End-to-End Cross-Layer Profiling Study of Congestion Control in High BDP Wireless Networks,” in *Proc. of the IEEE WCNC, 2007*, Mar. 2007.