A Conditional Retransmission Enabled Transport Protocol for Real-Time Networked Control Systems

Li Gui, Yu-Chu Tian and Colin Fidge
Faculty of Science and Technology
Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001, Australia
Email: {y.tian,c.fidge,l.gui}@qut.edu.au

Abstract—Real-time networked control systems (NCSs) over data networks are being increasingly implemented on a massive scale in industrial applications. Along with this trend, wireless network technologies have been promoted for modern wireless NCSs (WNCSs). However, popular wireless network standards such as IEEE 802.11/15/16 are not designed for real-time communications. Key issues in real-time applications include limited transmission reliability and poor transmission delay performance. Considering the unique features of real-time control systems, this paper develops a conditional retransmission enabled transport protocol (CRETP) to improve the delay performance of the transmission control protocol (TCP) and also the reliability performance of the user datagram protocol (UDP) and its variants. Key features of the CRETP include a connectionless mechanism with acknowledgment (ACK), conditional retransmission and detection of ineffective data packets on the receiver side.

I. INTRODUCTION

With the increasing integration of industrial processes and the rapid development of computer and network technologies, networked control systems (NCSs) are becoming increasingly significant [1]. Network-induced delay and packet dropouts are the main challenges in NCSs [6]. When wireless networks such as 802.11 are applied as replacements for hard-wired ones in an NCS to form a wireless NCS (WNCS), a larger amount of dropped data, and unpredictable and longer transmission latencies are introduced in the communication network due to the characteristics of wireless channels [5], [4]. These problems may degrade the NCS’s performance significantly or even cause system instability if the applications in the NCS are real-time control applications.

Real-time control is a class of time-critical applications which suffer more than other systems from packet losses, and unpredictable or long transmission delays. When applied to a real-time NCS, the most popular transport protocols, the transmission control protocol (TCP) and the user datagram protocol (UDP), behave with poor latency performance for TCP and unreliable communications for UDP. As a result, the NCS’s performance will degrade significantly or, even worse, the system will become unstable.

In this paper, a new transport layer protocol is presented which is based on UDP but provides a reliable delivery service. It employs the concept of retransmission and acknowledgment (ACK) while keeping data timeliness in mind. Retransmission is one of the most effective ways of compensating for lost packets. However, it causes unpredictable latency of packet delivery and consumes extra resources such as processor time, memory and network bandwidth. Therefore, the new protocol enables retransmission conditionally so that a trade-off between delivery reliability, transmission latency and consumption of network resources can be achieved.

II. LOGICAL DESIGN OF THE CRETP

The CRETP is a UDP based protocol with the ability to conditionally retransmit unacknowledged packets at the transport layer. It inherits UDP’s connectionless service to keep the protocol simple. In order to provide reliable network transmissions in an NCS, CRETP adds a sequence number to each data packet and enables acknowledgment for every successful data transmission. Since this protocol is designed for NCSs, the reliable transmission of a packet not only means the successful receipt of a packet at the receiving node, it also implies that the received packet is effective. A packet is considered effective if and only if it does not fail the checksum and it is not out-of-date.

A. Conditional Retransmission Mechanism

The key feature of the CRETP is its conditional retransmission mechanism. Since the CRETP is designed for real-time data transmissions in NCSs, it must achieve data timeliness while providing reliable transmissions. A unique feature of real-time control systems is the predominantly periodic traffic pattern, and the traffic load for each control loop is known in advance with a fixed control frequency [1]. A data packet becomes useless when a new data packet is available at the start of a new control period. Therefore, retransmission of the current data packet has to be stopped when a new control period starts and the new data packet is ready to be transmitted. This is the constraint on retransmission in the CRETP.

Like other retransmission enabled protocols, CRETP employs a retransmission timer that handles the waiting time for an ACK of a data packet. When choosing the timeout value of a retransmission timer, CRETP adopts TCP’s method of handling timeout and retransmission with our own tuned
parameters. In TCP, the round trip time (RTT) and timeout [2] are estimated as $RTT_{\text{new\_est}} = \alpha \times RTT_{\text{old\_est}} + (1 - \alpha) \times RTT_{\text{current}}$; Retransmission time = $\beta \times RTT_{\text{new\_est}}$, where $\alpha$ and $\beta$ are two constant weighting factors. Considering the requirements of NCSs, we recommend that the value of $\alpha$ be set in the range between 0.8 and 0.9, and the value of $\beta$ be set in the range between 1.5 and 2.

For updating $RTT_{\text{new\_est}}$, a timer back-off scheme is applied along with Karn’s algorithm [2] in the CRETP. The timer back-off strategy is widely used in retransmission enabled protocols so that increased delays due to retransmissions can be detected. In our simulations, the maximum of a back-off value is 6 times that of the present $RTT$ by constraining the maximum number of retransmissions to be 3 for each packet.

B. Data Effectiveness Detection Mechanism

Not all the packets that reach the receiver will be accepted by the destination CRETP node at the transport layer. Effectiveness of an arrived packet is detected before the CRETP conducts further actions.

There are several processes involved in data effectiveness detection. Firstly, the received packet should not fail the CRETP checksum. Secondly, when a packet successfully goes through the checksum, the destination CRETP checks its sequence number. An expired packet has to be detected and rejected even if it does not fail the checksum. The sequence number in a data packet header is the identifier that reveals the age of the data. Through comparing the sequence numbers, expired, out-of-date and duplicated packets can be detected and, as a consequence, discarded.

C. Acknowledgment Mechanism

The CRETP uses the ACK method to confirm the arrival of an effective data packet at the receiving end. The destination CRETP generates a corresponding ACK packet for each effective data packet it receives. A CRETP ACK can be designed as a CRETP packet with zero bytes of data and a flag value of 1 in its header, which is designed to be a fixed size of 16 bytes consisting of src port no. (2 bytes), dest port no. (2 bytes), total length (2 bytes), checksum (2 bytes), seq no. (2 bytes), flag (2 bytes) and timestamp (4 bytes). The timestamp field in the ACK header provides key information for $RTT$ calculation at the source CRETP. When an effective data packet is accepted by the destination CRETP, the value in its timestamp field is copied and saved in the timestamp field of its corresponding ACK packet.

The source CRETP will also check the effectiveness of the receiving ACK. An ACK is only effective when its sequence number equals the sequence number of current data packet in source CRETP.

D. State Transitions of the CRETP

While various tools have been employed in system analysis of the the logic design of the CRETP [3], finite state machines are presented in this paper to demonstrate the state transitions of the CRETP. They are depicted in Figure 1 for the source and destination CRETP, respectively.

III. PERFORMANCE EVALUATION

In order to provide a comprehensive performance evaluation for the CRETP, the CRETP was implemented in Network Simulator Version 2 (NS-2). Then, extensive simulations of the CRETP were conducted under various typical scenarios. The detailed design for the CRETP implementation in NS2 can be found elsewhere [3].

Evaluation of the CRETP was conducted through comparative studies for the behaviour of all three protocols, UDP, TCP and CRETP. This is because we intend to test if the CRETP can (1) perform better than UDP in NCSs in terms of transmission reliability; and (2) improve the delay performance over TCP to an acceptable level for NCSs. Only if these two questions get positive answers can we claim that the CRETP has the ability to greatly improve the transmission reliability and keep data timeliness for real-time applications in NCSs. It will also mean that the CRETP distinguishes itself among the three protocols by showing the best overall performance for real-time control systems.

A. Network Specifications

In a WNCS, the number of sensors, controllers, actuators, and other devices interconnected within a physical subtask is
typically low, e.g., a few tens or less [1]. This is different from other general wireless network applications. The maximum number of communicating devices in our simulation scenarios was set to be 11. In our evaluation, twenty-six scenarios were simulated altogether. For consistency of the comparative studies, all scenarios shared some basic settings. Table I lists basic configurations of the wireless network model that were used in all scenarios. In addition to the same basic wireless model, all the simulated scenarios have some common presumptions specified as follows: 1) All sensors and the controller are fixed in the network field with sensors distributed in a circle around the controller, which implies that distances between the controller and sensors are the same and constant as the circle’s radius is always 50 metres; 2) All sensors start their communications with the controller at 1s and stop at 16s; 3) Sensors are designed to generate a control data packet at the beginning of each control period; 4) The traffic type on the application layer in all sensors is Constant Bit Rate (CBR); and 5) The application data has a fixed size of 200 bytes.

### TABLE I
**WIRELESS MODEL USED IN SIMULATIONS**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Network standard:</td>
<td>IEEE 802.11b</td>
<td></td>
</tr>
<tr>
<td>Radio channel data rate:</td>
<td>1.0 Mbps</td>
<td></td>
</tr>
<tr>
<td>Network area:</td>
<td>250m × 300m square</td>
<td></td>
</tr>
<tr>
<td>Radio-propagation model:</td>
<td>Two-ray ground</td>
<td></td>
</tr>
<tr>
<td>Routing protocol:</td>
<td>Dynamic Source Routing</td>
<td></td>
</tr>
<tr>
<td>Wireless interface (MAC) buffer type:</td>
<td>Drop-tail priority queue</td>
<td></td>
</tr>
<tr>
<td>AntennaD:</td>
<td>OmniAntenna</td>
<td></td>
</tr>
</tbody>
</table>

### B. Case Study One

As a major factor in network protocol performance, the channel condition was employed as the only variable parameter to draw distinctions between the 11 simulation scenarios in the first case study. The difference between these wireless conditions was the number of irregular run-time channel errors happening during the simulation. The number of sensors was 5 while the control period was 50ms.

When UDP was used in Case One, the average delays were limited within 10.09ms, regardless of the channel’s condition. As for CRETP, the average delays were just slightly longer than those provided by UDP, with an upper bound of 13.31ms. The average end-to-end delays for all successfully received data packets are shown graphically in Figure 2. It can be seen that, for UDP or CRETP, the average delays are almost constant as the network’s condition gets worse. TCP behaves the worst with much longer time delays.

When transmission reliability is considered, UDP’s performance degrades. Moreover, when data effectiveness is taken into account, TCP does not provide the most reliable transmission for real-time applications. Table II summarizes the percentage of effective data packets received at the controller in different scenarios. No matter what transport protocol is used, the ratio of effective data decreases as the network’s condition deteriorates. CRETP’s conditional retransmission scheme enhances communication reliability while keeping data timeliness in mind, which means it always had the highest value of percentage of effective data packets among the three protocols.

Overall, in this case study, comparative studies of the performance of the three protocols demonstrates that the CRETP best satisfies the requirements of data timeliness as well as transmission reliability for real-time applications in an NCS over wireless networks that are vulnerable to errors.

### C. Case Study Two

As another major factor influencing protocol performance, traffic load was considered as the parameter in this case study. The wireless channel condition did not change in all scenarios. We tested the protocol performance in nine scenarios as shown in Table III. Scenarios 1 to 3 used the same control period of 70ms, while the control period in Scenarios 4 to 5 was 80ms. In Scenarios 7 to 9, the control period was set to be 90ms. For each of the three control periods, the number of sensors had three different values. The minimum value was 5, the same as that in Case Study One. The medium and maximum values were 8 and 10, respectively.

Table IV records the average end-to-end delays for all successfully received data packets in the different scenarios when using different transport layer protocols. It can be seen that UDP still guarantees the smallest average delay while TCP introduces the longest. CRETP performs not as well as UDP with respect to delays; however, its average delays are...
The simulation results in this case study showed that the CRETP performs the best among the three protocols when both transmission delay and reliability are taken into account. This conclusion is the same as in Case Study One. New information found in this case study was that although all protocols performed worse when more working sensors were put into the network, the CRETP can recover and even improve its reliability if a longer control period is applied.

After analyzing the simulation results in the above two case studies, the questions posed at the beginning of Part III can be positively answered. Comparative studies demonstrate that only the CRETP can guarantee data effectiveness of compensated packets and keep the delay performance at an acceptable level at the same time. These advantages make the CRETP a proper transport protocol which can greatly improve the overall performance of real-time NCS applications.

IV. Conclusion and Future Work

A new transport layer protocol, CRETP, has been developed in this paper for real-time WNCS applications. It employs a conditional retransmission mechanism to significantly improve data transmission reliability. Unacknowledged data packets will be retransmitted by the CRETP for a certain amount of time to compensate for data losses. As every data packet in real-time control systems is useful only within a certain deadline, the CRETP has the ability to check data effectiveness and guarantee that every data packet delivered to the application layer is valid. The performance of the CRETP has been evaluated through simulations with comparative studies between the CRETP and two commonly-used transport protocols, UDP and TCP. Further developments in this area are being carried out to implement the CRETP in Linux for practical WNCS applications.

ACKNOWLEDGMENT

This work was supported in part by the Australian Government’s Department of Innovation, Industry, Science and Research (DIISR) under International Science Linkage (ISL) Grant No. CH070083.

REFERENCES