Computing the Retransmission Timeout in CoAP

Ekaterina Balandina[1,2], Yevgeni Koucheryavy[2], and Andrei Gurtov[3]

[1] FRUCT Oy, Helsinki, Finland
Ekaterina.Balandina@fruct.org
[2] Tampere University of Technology, Tampere, Finland
yk@cs.tut.fi
[3] Department of Computer Science and Engineering, Aalto University, Helsinki, Finland
gurtov@hiit.fi

Abstract. The most prominent IT trend nowadays is connection of Wireless Sensor Networks (WSNs) with Internet service infrastructure. Interconnection of the millions of sensor and processing devices will create a tremendous traffic increase that can lead to congestion. In parallel to the development of new protocols for WSNs, e.g., Constrained Application Protocol (CoAP) there is plenty of research for new congestion control techniques (CC). This research shall carefully take into account all key restrictions of sensor networks, e.g., memory and power consumption, lousy paths and limited links throughput. This paper analyzes classical approach of definition of the retransmission timeout (RTO) estimate, proposed in RFC 6298, and compares it with the Eifel Retransmission Timer and the new ideas proposed in CoCoAP. Finally, we present our method for calculating RTO. Our approach could be seen as an extension of the classical TCP algorithm, where instead of constants that are used to take into account history of the current state we use a dynamically changing parameter. The value of this parameter is defined as a ratio between current sample of the round-trip time (RTT) and the RTO value.

Keywords: WSNs, Congestion Control, CoAP, RTO.

1 Introduction

Wireless sensor networks (WSN) are a fast growing technological domain and quite common phenomenon nowadays. Different types of WSNs are deployed for provision of a numerous services in various application areas, e.g., medicine, industrial, civil and army areas, etc [1]. The main challenge of WSN infrastructure is that most devices (sensors) in such networks have very limited memory, power and processing resources [2] and as a result require special resource-efficient software to be executed on top of them.

One of the key trends in today's ICT research is development of an efficient infrastructure for connecting WSNs with Internet services. It is important to mention here that most of classical Internet services are done without proper energy-awareness. At the same time sensor networks are characterized by short lifetime of sensors, inability to properly handle redundant traffic and network overloads. As a result, direct deployment of such services on sensor platforms will lead to unacceptable waste of energy, which cannot be delivered on the sensor side [3]. Moreover, complexity of Internet infrastructure and amount of network traffic will grow by orders of magnitude.

Constrained Application Protocol (CoAP) was proposed to address these demands [4]. It is light and efficient enough to work on constrained devices and provides good interface for the standard Internet services. CoAP protocol works on top of the unreliable UDP transport layer. A recent CoAP version has just a simple back-off mechanism that includes a timer and a retransmission counter. This is not sufficient for proper congestion management and there is a need in reliable congestion control mechanism that will address most of network overload scenarios.

In fact, high probability of congestion is in very core of most typical WSN use case scenarios. Service reliability in WSN is often based on sensing redundancy, when multiple sensors producing similar information in the same time and place. For example, in a forest fire and area lightening detection applications most likely a number of sensors will at the same time detect an event and start to signal about it. This may result in network overload for this network segment and even loss of important messages due to mutual blocking.

An important WSN design aspect is organization of sensor-scheduling activities that guarantee full area coverage while maximizing network lifetime [5]. Good overview of this problem with solution on how ensures full coverage of the monitored area by involving minimum number of sensors, which minimizes energy consumption and therefore extends network lifetime, was done by Lehsaini at el. [6]. Another key issue is efficient organization of data routing in WSNs [7].

However, proper control of network load cannot be provided without use of a congestion management mechanism [8]. Many approaches can be employed for this purpose [9], [10, [11]. In this study we decided to explore one of the most classical ideas, i.e., control congestion by choosing a policy for recalculating the retransmission timeout (RTO) values, which help to keep the transmissions on the required target level and also allows stopping transmission of flows when the network cannot deal with pushed amount of traffic. RTO is the time that elapses after a packet has been sent until the moment when sender will consider it to be lost and therefore a retransmission shall be initiated. The RTO could be seen as a prediction of the upper boundary of the round-trip time (RTT). Development of a method for accurate RTO prediction will provide the congestion control mechanism with a key tool to drive network via events of heavy load with minimal service degradation. It greatly influences reliable end-to-end performance. To evaluate importance of RTO it is enough to consider two opposite polices of recalculating RTO. A spurious timeout - too optimistic retransmission time can cause unnecessary traffic, which is reducing connection’s effective throughput. A conservative retransmission timer causes long idle times before the lost packet is retransmitted during timeout period sensor is active and in vain spending energy.

The rest of the paper is structured as follows. Section II provides an overview of the Constrained Application Protocol. In Sections III and IV we discuss the new method for RTO calculation. The developed analytic model of the new RTO calculation methods is presented in Section V. Section VI gives analysis and evaluation of RTO behavior in various scenarios. The paper is concludes by the list of main conclusions, acknowledgments and list of references.

2 Constrained Application Protocol (COAP)

Constrained Application Protocol was proposed by IETF group and in October 2012 its 12th draft was published. CoAP is a specific web protocol which is developed specifically for constrained nodes (sensors) and low-power networks that have high packet error rates and relatively small throughput (6lowPAN).

CoAP is designed specifically for machine-to-machine (M2M) applications that have embedded multicast support asynchronous message exchange and low overhead. Design of the interaction model of CoAP is very similar to the client/server model used in HTTP, but taking into account specifics of M2M applications, CoAP nodes can act either as clients or servers.

The send rate of CoAP protocol can be defined using a simplified version of TCP send rate formula, which has been derived by Padhye at el. in [12]. But for COAP case values of W_M and b are always equal to 1, so these parameters shall be excluded from calculation.

Unlike HTTP, the internal communication of CoAP is based on asynchronous datagram-oriented UDP transport layer. But it is important to remember that UDP does not provide internal mechanisms for congestion management and this is why one of the key requirements to ensure stable work of CoAP is to have own Congestion Control mechanism (CC). In this paper we propose an enhancement of the classical CC method, which is specifically adopted for CoAP. Current draft of CoAP specification states that CoAP has mechanisms for slowing down network overload, correct order of packets and check duplicates by using the exponential back-off mechanism and simple stop-and-wait mechanism [4]. This is achieved by strictly limiting number of simultaneous outstanding interactions to one, where the outstanding interaction is a confirmable message (CON) for which the acknowledgement (ACK) has not yet been received but is still expected. The second allowed case is when there was a request, for which neither a response nor ACK has yet been received, but is still expected. In fact both cases could occur at the same time, which is counted as one outstanding interaction. Message duplication detection is implemented for both confirmable and non-confirmable messages based on a simple idea of including message identification field to the message header and definition of the recipient endpoint.

Two parameters can be used for controlling the back-off mechanism - retransmission timeout (RTO) and retransmission counter. The initial value of retransmission timeout in CoAP is set to a random number within the interval of [ACK_TIMEOUT to ACK_TIMEOUT*ACK_RANDOM_FACTOR], where ACK_TIMEOUT and ACK_RANDOM_FACTOR are the transmission parameters, which default values are 2 sec. and 1.5 respectively. The initial value of the retransmission counter is always 0. The maximum number of retransmissions is defined by parameter

MAX_RETRANSMIT and its default value is 4. The retransmission timer switch-on when CON has been sent, and the timer value doubles each time, when the timer expires and no ACK for the CON had been received. After four not successful attempts of retransmission, the sender shall close the session.

The following parameters are used for controlling the retransmission time in the current CoAP draft [4]:

- MAX_TRANSMIT_SPAN - is the maximum time from the first transmission of a confirmable message to its last retransmission.

$$\text{MAX\_TRANSMIT\_SPAN} = \text{ACK\_TIMEOUT} * (2^{\text{MAX\_RETRANSMIT}} - 1) * \text{ACK\_RANDOM\_FACTOR} \quad (1)$$

Default value for MAX_TRANSMIT_SPAN is 45 seconds.

- MAX_TRANSMIT_WAIT - is the maximum length of period of time that sender is capable to wait for ACK on the already sent confirmable message.

$$\text{MAX\_TRANSMIT\_WAIT} = \text{ACK\_TIMEOUT} * (2^{\text{MAX\_RETRANSMIT} + 1} - 1) * \text{ACK\_RANDOM\_FACTOR} \quad (2)$$

Default value for MAX_TRANSMIT_WAIT is 93 seconds.

- MAX_LATENCY - is the maximum time that will be needed for a datagram to be fully delivered to the destination. By default it is set to be 100 seconds.

- PROCESSING_DELAY - is the time required for a node to process a confirmable message and issue an acknowledgement. By default it is equal to ACK_TIMEOUT.

- MAX_RTT - is the maximum round-trip time calculated as follows:

$$\text{MAX\_RTT} = 2 * \text{MAX\_LATENCY} + \text{PROCESSING\_DELAY} \quad (3)$$

- EXCHANGE_LIFETIME - is the time from the moment when transmission of the confirmable message has started until the moment when an acknowledgement is no longer expected. As a result at the message layer information about this message exchange can be purged.

$$\text{EXCHANGE\_LIFETIME} = \text{ACK\_TIMEOUT} * (2^{\text{MAX\_RETRANSMIT}} - 1) * \text{ACK\_RANDOM\_FACTOR} + 2 * \text{MAX\_LATENCY} + \text{PROCESSING\_DELAY} \quad (4)$$

For the default transmission parameters its value is 247 seconds.

Responsibility for congestion prediction, detection and control in CoAP networks is fully on clients' side. But potentially it is possible that the client will be hacked or broken, which will lead to abnormal behavior. To prevent any damage to the network in such cases, the server should have some mechanism that will limit the traffic data rate of nodes with abnormal behavior and this solution is discussed in the next section.

3 Enhancement of the Classical TCP Algorithm

The most classical definition of the algorithm for calculating value of the retransmission timer is proposed by RFC 6298 [13]. According to that definition the Transmission Control Protocol (TCP) uses an RTO to control reliability of data exchange. Calculation of new RTO value is done by an algorithm that is based on two variables: smoothed RTT (SRTT) and RTT variation (RTTVAR). SRTT can be seen as a mean to preserve history of RTT, its impact factor is constant and equals to 7/8. RTTVAR keeps the history of RTT variation, it is also constant and its impact factor is 3/4.

Before the first measurement of RTT is received RTO should be set to 1 second. After first RTT sample is received the following formulas are used for RTO calculation:

$$\begin{aligned} SRTT &\leftarrow RTT \\ RTTVAR &\leftarrow RTT/2 \\ RTO &\leftarrow SRTT + \max(G, K{*}RTTVAR) \end{aligned} \tag{5}$$

After subsequent measurement is received the following formulas are applied:

$$\begin{aligned} RTTVAR &\leftarrow (1 - \beta) * RTTVAR + \beta * |SRTT - RTT| \\ SRTT &\leftarrow (1 - \alpha) * SRTT + \alpha * RTT \\ RTO &\leftarrow SRTT + \max(G, K{*}RTTVAR) \end{aligned} \tag{6}$$

In these equations α, β and K are constants and their values respectively are 1/4, 1/8 and 4. Value G defines the clock granularity and higher it is, more conservative is result RTO value. It is recommended to choose G value not greater than 100 ms [13]. At the same time G should be at least one order of magnitude smaller than the RTT [14].

In the later study we will primarily address the Machine-to-Machine (M2M) use case scenario. Shafiq at el. [15] illustrate the cumulative distribution functions (CDFs) of the median RTTs and packet loss ratio experienced by each device for smartphones and all M2M device categories. Based on this study we can say that if RTT varies between 500 ms and 2 seconds, G should be at least 50 ms and its maximum value in this case is 100 ms, to fulfill recommendation of RFC6298 and [15].

In RFC6298 it is underlined that SRTT and RTTVAR can be cleared if retransmission timeout expired several times and values of these variables became bogus.

4 CoCoAP Modifications

In addition to activities targeted in improvement of CoAP that resulted in releasing specification draft version 12, the same team proposed to take RTO estimation calculation algorithm [14] as a basis for an enhanced protocol solution on top of CoAP. The new protocol is named CoAP Simple Congestion Control/Advanced (CoCoAP).

The initial RTO estimate in this protocol is set to 2 seconds. Modification of RTO value is done by use of two mechanisms named “strong” and “weak” estimators. Both mechanisms implement the same algorithm, but have different sets of state variables. If the packet is received based on the initial transmission, i.e., without any retransmissions, then the “strong” estimator RTO calculation branch is in use. If the last packet was received as retransmissions were done then the “weak” estimator branch is used. It means that if we don’t know for sure whether ACK is an acknowledgment of the initial message that was just delayed or it is already an acknowledgement of the retransmitted packet then the “weak” estimator is in use. If ACK came before retransmission timer expires, it means that real RTT sample was calculated and in this case “strong” estimator branch of RTO calculation algorithm shall be used. The last step is overall RTO estimate calculation that is an average of the currently calculated “weak” or “strong” value and the RTO overall value obtained on the previous step.

$$\text{RTO\_overall} = 0.5*\text{RTO\_recent} + 0.5*\text{RTO\_overall} \tag{7}$$

CoCoAP support service provision for packets that don’t need confirmation of delivery. Handling of such packets is provided by advanced part of the algorithm that defines a set of additional rules. For example, the date rate for sending non-confirmable messages must not exceed 1 Byte/s and at least 2 out of 16 consecutive messages sent to one endpoint must be confirmable. The full set of additional rules can be found in CoCoAP draft specification [14].

5 Analytic Model of Send Rate

The send rate is a key characteristic of network quality and it strongly dependents on frequency of congestion events as well as on speed of transmission recovery after such events. In this section we present derivation of an analytical characterization of CoAP send rate as a function of loss rate and RTT. This part of the study was partly inspired by work of Padhye at el. [12].

CoAP protocol detects packet loss when the retransmission timeout is expired, which in case of proper configuration of RTO happens if a packet or the corresponding ACK is lost. For simplicity reasons and to make it easy to read and understand the first steps of derivation process we adopting terminology and notations proposed in [12].

We would like to start from a general formula of send rate B_t. For any given time $t > 0$ and N_t - number of packets transmitted during interval t, we can define the send rate as $\mathrm{B}_\mathrm{t} = N_\mathrm{t}/t$ (the number of packets sent per unit of time).

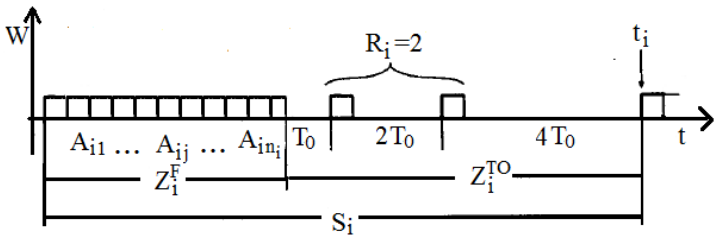

Figure 1. CoAP performing scheme with timeout loss indications

As it was underlined before in Section II, CoAP sends one packet and waits for the ACK. As a consequence it is enough for CoAP to support only one type of loss indication, which is based on occurrence of timeout event TO. $Z^F{}_i$ is the time interval of normal transmission without packet loss. In practice it consists of n_i RTT intervals, as it is illustrated in Figure 1 by A intervals. This continues until the first timeout event (TO) occur. Then the protocol starts retransmission process, which on the sender side is associated to the sequence of timeouts $Z^{TO}{}_i$. Based on these notions the first full cycle of normal transmission and recovery for an error is defined as S_i:

$$S_i = Z^{TO}{}_i + Z^F{}_i \tag{8}$$

The number of packets sent during $Z^{TO}{}_i$ is denoted as R_i. It counts the total number of packet transmissions in $Z^{TO}{}_i$. Parameter n_i is a number of packets sent during $Z^F{}_i$. Then we can define the total number of packets sent during S_i period (M_i) as follows:

$$M_i = n_i + R_i\, Z^F{}_i \tag{9}$$

So the most general definition of send rate B could be defined as a relation function of the total number of packets send (M) to the transmission periods (S).

$$B = \frac{E[M]}{E[S]} = \frac{E\left[\sum_{j=1}^{n_i} A_{ij}\right]+E[R]}{E\left[\sum_{j=1}^{n_i} A_{ij}\right]+E[Z^{TO}]} \tag{10}$$

Let’s denote the average value of RTT as E[r]. Obviously when only one packet is allowed to be transmitted, the average value of A intervals denoted as E[A] is:

$$E[A] = E[r] \tag{11}$$

Based on formulas 10 and 11 we can derive the the result formula for the send rate:

$$B = \frac{RTT+E[R]}{RTT+E[Z^{TO}]} \tag{12}$$

Timeout occurs *k* times for *k* - 1 consecutive losses, where the number k of timeout occurrences has a geometric distribution

$$P[R=k] = p^{k-1}(p-1) \tag{13}$$

Then the mean of R could be defined as:

$$E[R] = \sum_{k=1}^{n} P[R=k] = 1/(1-p) \tag{14}$$

The maximum number of retransmissions in CoAP is four. The duration of the sequence of four timeouts is denoted as L, where T_0 is the initial value of retransmission timer.

$$L = (2^4 - 1)T_0 \tag{15}$$

Now we can define the average time interval of timeout transmission after packet loss was detected: $E[Z^{TO}]$

$$E[Z^{TO}] = \sum_{k=1}^{4} L_k P[R=k] = T_0(1+2p+4p^2+8p^3-15p^4) \tag{16}$$

So finally we can calculate the send rate of CoAP protocol as:

$$B(p) = \frac{RTT+1/(1-p)}{RTT+T_0 f(p)} \tag{17}$$

where $f(p) = T_0(1+2p+4p^2+8p^3-15p^4)$

6 Evaluation

The proposed enhancement of CoCoAP protocol is based on combination of RTO estimation algorithm [13] and a set of the algorithm enhancements proposed in [14]. Also here we have used main findings of a survey of the current works involved in traffic analysis and modeling, network optimization and network anomaly detection for WSNs [16]. For our studies we also used the results by Ponmagal and Ramachandran [17] on wireless rate-control technique, whose link characteristics are identified by a variable link rate and burst transmission error. In addition the proposed CoCoAP enhancement utilizes ideas from the Eifel retransmission timer [18], [19].

The study of Eifel retransmission timer made an important conclusion on a role of "magic numbers" (α, β and K) for the performance of the algorithm. Estimator gains (values of α and β) are too high and the variation weight (K) is too low for the situations of the large senders load. They cause SRTT and RTTVAR to decay too quickly and RTO becomes too aggressive [14]. Based on these conclusions we propose to replace constant coefficients α and β to a coefficient γ that is defined as follows:

$$\gamma = RTT/RTO \tag{18}$$

As it was shown in the previous section, in current CoCoAP [13] RTO_overall is defined as an average of RTO recent (that is based on currently calculated "weak" or "strong" value) and RTO overall obtained on the previous step (see formula 7). In the proposed enhancement this rules works only if RTO recent was calculated based on "weak" estimate. In cases when RTO recent was calculated based on "strong" estimate then we assign RTO_overall = RTO_recent.

Figure 2 and Figure 3 compare the performance of the classical RTO estimation algorithm (blue dotted line), modified RTO estimation algorithm (red line) and the stair-step RTT function or saw-like RTT function respectively (green line).

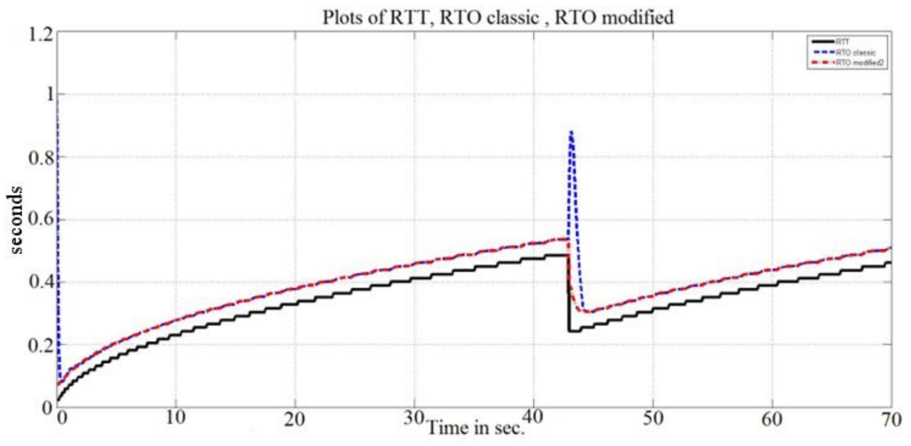

Figure 2. Plots of the classical RTO estimation algorithm (blue), a modified RTO estimation algorithm (red) and with stair-step RTT function (black)

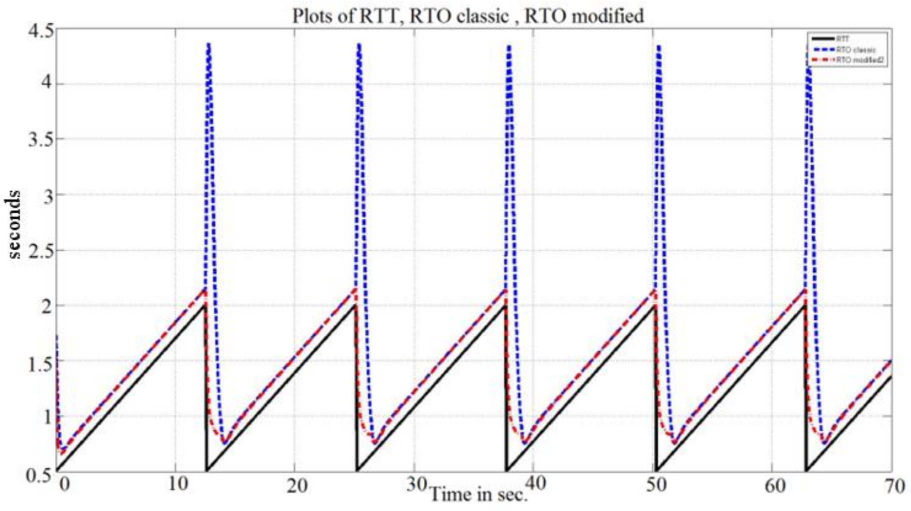

Figure 3. Plots of the classical RTO estimation algorithm (blue), modified RTO estimation algorithm (red) and with a saw-like RTT function (black)

This result is based on intensive simulation and illustrates the same level of performance. In Figure 2, we can see a peak value for the classic RTO algorithm after approximately 42 seconds of simulation time. This peak is related to the decrease of RTT value, which first leads to a peak increase of RTO value as we can see. This happens as a difference of RTT and SRTT becomes negative, but formula (6) is calculating RTTVAR value using modulus that results in such an artificial increase of the RTO value. The modified algorithm adapts to this situation and reacts respectively by decreasing RTO prediction value [18].

Based on an intensive simulation studies, the following recommendations were generated. The first recommendation is that if RTT – SRTT < 0 then it is beneficial to skip the rule of changing RTTVAR, as it will allow faster adaptation to the current RTT values, while if RTT – SRTT > 0 we use the standard formula for changing RTTVAR as defined in [14]. The second recommendation is that when $\gamma > 0.5$ then it is better to change its value to $1-\gamma$, as it means RTT is greater than ½ RTO, which could be only a consequence of the previous packet lost and it is better to slowdown adoption mechanism to avoid fast return to the too high level of data transmission.

7 Conclusion

The paper gives an overview of the existing congestion control solutions for wireless sensor networks, the CoAP protocol, and proposes the enhancement of CoCoAP protocol that combines best features of the prior art solutions and provides a set of new rules. The set of new rules is generated based on the results of intensive simulation tests that allowed us to create a new RTO estimation algorithm that works significantly better than the classical one. Our algorithm is slightly more aggressive, but at the same time more efficient in conditions of limited traffic fluctuations, as well as when some unpredictable events occur.

The performed analysis allowed us to derive an analytical model for CoAP congestion control behavior by defining the sending rate as a function of loss and RTT. In this model we use the COAP timeout mechanism as a tool of detecting packet losses.

Currently we are working on development of a more sophisticated model of the proposed algorithm. The implementation is targeted for Cooja simulator [20] that provides a good platform for this kind of study. We also plan to evaluate the new RTO mechanism using trace-driven simulations on real network data.

Acknowledgements: Authors are grateful for TiViT IoT SHOK program and GETA scholarship program of Tampere University of Technology that provided required support of this research. Also we want to acknowledge support MAMMOTH and SEMOHealth projects.

References

1. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci,“Wireless sensor networks: a survey”, Computer Networks, Elsevier, Vol. 38, Issue 4, pp. 393-422. March 2002.
2. M. Cardei, M.T. Thai, Y. Li, and W. Wu, “Energy-efficient target coverage in wireless sensor networks”, IEEE INFOCOM’05, pp. 1976-1984, 2005.
3. Y. Cai, M. Li, W. Shu, and M.Y. Wu, “Acos : A precise energy-aware coverage control protocol for wireless sensor networks”, International Journal Ad Hoc Sensor Wireless Networks, Vol. 3, Issue 1, pp. 77-98, 2007.
4. Z. Shelby, K. Hartke, C. Bormann, and B. Frank, “Constrained Application Protocol (CoAP)”, draft-ietf-core-coap-12, October 2012.
5. R. Mulligan and H.M. Ammari, "Coverage in Wireless Sensor Networks: A Survey", Network Protocols and Algorithms, Macrothing Institute, ISSN 1943-3581, Vol. 2, No. 2, pp. 27-53, 2010.
6. M. Lehsaini, H. Guyennet, and M. Feham, "Cluster-based Energy-efficient k-Coverage for Wireless Sensor Networks", Network Protocols and Algorithms, Macrothing Institute, ISSN 1943-3581, Vol. 2, No. 2, pp. 89-106, 2010.
7. S. Ganesh and R. Amutha, "Efficient and Secure Routing Protocol for Wireless Sensor Networks through Optimal Power Control and Optimal Handoff-Based Recovery Mechanism", Journal of Computer Networks and Communications, Vol. 2012, URI: http://www.hindawi.com/journals/jcnc/2012/971685/, 2012. Retrived on 08.05.2013.
8. A. Gurtov, TCP Performance in the Presence of Congestion and Corruption Losses, Master's Thesis, University of Helsinki, Department of Computer Science, December 2000.
9. L. Chen, B.K. Szymanski, and Joel W. Branch, "Auction-Based Congestion Management for Target Tracking in Wireless Sensor Networks", In Proc. of the 7th IEEE PERCOM Conference, pp. 194-203. March 2009.
10. B. Kumara and M.M. Naik, "Architecture for Node-level Congestion in WSN using Rate Optimization", IOSR Journal of Engineering, ISSN 2250-3021, Vol. 2, Issue 6, pp. 30-34, URI: http://www.iosrjen.org/ Papers/vol2_issue6%20%28part-3%29/E0263034.pdf, June 2012. Retrived on 08.05.2013.
11. F.K. Shaikh, A. Khelil, A. Ali, and N. Suri, "Reliable congestion-aware information transport in wireless sensor networks", International Journal of Communication Networks and Distributed Systems, Vol. 7, No 1/2, pp. 135-152, 2011.
12. J. Padhye, V. Firoiu, D.F. Towsley, and J.F. Kurose, "Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation", IEEE/ACM Transactions on Networking, Vol. 8, No. 2, pp. 133-145, April 2000.
13. V. Paxson, M. Allman, J. Chu, and M. Sargent, “Computing TCP's Retransmission Timer”, RFC 6298, June 2011.
14. C. Bormann, “CoAP Simple Congestion Control/Advanced”, draft-bormann-core-cocoa-00, August 2012.
15. M. Z. Shafiq, L. Ji, A.X. Liu, J. Pang, and J. Wang, “A First Look at Cellular Machine-to-Machine Traffic - Large Scale Measurement and Characterization”, SIGMETRICS’12, London, UK, June 2012.
16. Q. Wang, "Traffic Analysis & Modeling in Wireless Sensor Networks and Their Applications on Network Optimization and Anomaly Detection", Network Protocols and Algorithms, Macrothing Institute, ISSN 1943-3581, Vol. 2, No. 1, pp. 74-92, 2010.
17. R.S. Ponmagal and V. Ramachandran, "Link Quality Estimated TCP for Wireless Sensor Networks", International Journal of Recent Trends in Engineering, Vol. 1, No.1, pp. 495-497, May 2009.
18. R. Ludwig and K. Sklower “The Eifel Retransmission Timer”, ACM SIGCOMM Computer Communication Review, Vol. 30, Issue 3, pp. 17-27, July 2000.
19. R. Ludwig, and A. Gurtov, The Eifel Response Algorithm for TCP, RFC 4015, February 2005.
20. Official web page of Cooja simulator in Contiki OS, URI: http://www.contiki-os.org/. Retrived on 08.05.2013.