International Journal of Future Generation Communication and Networking
Vol. 9, No. 7 (2016), pp. 123-134
http://dx.doi.org/10.14257/ijfgcn.2016.9.7.12

Performance Analysis of Reprogramming Algorithms in Wireless Sensor Networks

Kire Jakimoski[1], Sime Arsenovski[2], Slavcho Chungurski[3] Oliver Iliev[4]

[1,2,3,4]*Faculty of Informatics, FON University, Skopje, Republic of Macedonia*
[1]*kire.jakimoski@fon.edu.mk;* [2]*sime.arsenovski@fon.edu.mk;*
[3]*slavcho.chungurski@fon.edu.mk;* [4]*oliver.iliev@fon.edu.mk*

Abstract

Wireless Sensor Networks (WSNs) usually include a large number of small-sized battery powered sensor nodes that integrate sensing, computing, and communication capabilities. Nowadays, Wireless Sensor Networks are very attractive topic that is still in the state of active development. In this work conventional reprogramming agents of Wireless Sensor Networks are analyzed and performance and efficiency of existing reprogramming agents is researched. Suggestions about the modifications are also proposed to improve its performances. Performances of the Completion Time, Suppression and Energy Consumption for reprogramming protocols are examined and important conclusions are made. Results show that S2Torrent with Selective approach protocol gives the best results.

Keywords*: Deluge, energy consumption, reprogramming, wireless sensor networks, object size, overall completion time, suppression*

1. Introduction

Wireless Sensor Networks are very attractive worldwide because of its low cost, and easily affordable means to monitor environmental conditions. Wireless Sensor Network is a resource constrained network with active research in progress. There are still network reprogramming requirements. Embedded nature of sensor networks requires propagation of new codes over the network [1].

Wireless sensor network scales reaching thousands of nodes. So, there is a necessity in debugging and testing cycle. Software that is already installed in sensor nodes may need to be updated with new functionalities or features [2]. Therefore, parameters and settings may have to be changed over time.

Regarding to network reprogramming, it is very important the capability for remote reprogramming of sensor nodes via wireless network to be supported [3]. One of the main benefits is that it allows for over-the-air software updates in sensor networks. It also enables sensor nodes to be self-reprogrammed so they can adapt to changing tasks and evolving environments. WSNs deployment in remote areas often makes impossible the capability to physically retrieve them. Manual reprogramming of each node by plugging in the device to the computer or a PDA is also not feasible for large networks. Therefore, the requirement to reprogram a network of nodes through the wireless medium becomes a necessity.

Network reprogramming is also helpful to patch errors in the software after deployment. For example, a bug could be found in the deployed sensor operating system (OS) or a logical error in the WSN routing that results in undesirable looping [4]. However, in all mentioned cases, the actual software in sensor nodes will be needed to be updated.

Most of the current networks' reprogramming protocols are focused on propagating the same code image to a network of homogeneous sensor nodes [5]. Inexperienced

ISSN: 2233-7857 IJFGCN

approaches that adapt this kind of protocols for heterogeneity are largely inefficient. Ease of programming has long been recognized as a major hurdle to the adoption of WSN technology. In response to this need, several platform dependent programming solutions have been developed till now. However, a well-established characterization of the available approaches is largely missing. As a result of this, researchers are unable to orient themselves in this diverse field, and developers struggle in identifying the solutions that are most appropriate to their application requirements.

One of the most important challenges of ad hoc & sensor networks is the efficient and distributed control of the channel. Because the efficient and distributed control of the channel occurs at the MAC layer, researches are more focused on the MAC layer [6]. Reprogramming the sensor nodes is more economical and practical than deploying new sensor nodes. Advances in the field of technology have enabled availability of software patches for the applications through reprogramming.

In the work presented in this paper the overall completion time is observed for existing reprogramming protocol (Deluge) and the proposed reprogramming protocol with two approaches. Experimental setup and assumption is also done for reprogramming. Sensitivity analysis of the parameters of the proposed work is performed for their relative significance in terms of convergence, diversity and computational complexity.

The rest of this paper is organized as follows. Section 2 presents and overview of reprogramming protocols and challenges in ad hoc & sensor wireless networks. Section 3 describes the simulation parameters that were used for the obtained results of these researches. Simulation results are presented in Section 4, and Section 5 concludes the paper.

2. Overview of Reprogramming Protocols

When a sensor node is reprogrammed, first phase is the decision about the needed reprogramming and about the piece of the code that requests update. According to the application type, system user could initiate this phase or it could be done automatically by the nodes in a distributed manner. If the system user is doing this, reprogram command is issued by the system user together with a group of attributes. If this phase is initiated automatically by the nodes, a decision for update of the code is done by the nodes.

Deluge is one of the greatly utilized and identified protocols for reprogramming [7]. This protocol is built on previous efforts in density-aware protocols and numerous optimizations are involved in this case. It is important to note about Deluge that it includes splitting the code into a group of fixed-size pages. As a result of this spatial multiplexing is enabled where pages are dealt with as independent transfer objects.

Another type of reprogramming protocol is Stream [8] which is built on Deluge and it includes optimization of the sent information over the channel. SYNAPSE is also reprogramming system for Wireless Sensor Networks that is intended for improving the effectiveness of the error recovery phase [9]. Reprogramming protocol that is density-aware and light-weight is proposed by the authors in [10]. It is named ReXOR and it includes XOR encoding in order to decrease the communication cost.

In our work we present performance analysis of the Deluge and proposed reprogramming protocols.

S2Torrent is the proposed code propagation protocol in this research work. In order to simulate the enhanced routing agent Network Simulator OMNet++ is used in this work. S2Torrent uses Trickle's and Deluge's technique of periodic advertisements and adds support for distributing large objects to whole network or to the specific group of nodes in the network.

Encoding/decoding: Regarding to the subject of encoding/decoding, it disseminates the program code in an epidemic fashion to propagate the program code while regulating the excess traffic. S2Torrent similar to Deluge, divides every

program image into sizeable pages and each page is further divided into packets. It reads the new program image, encodes it into data packets, and sends packets out through the radio. Then, the receiver is decoding these packets and rebuilding the program image.

Regarding to the issue of single-hop/multihop reprogramming, S2Torrent supports multi-hop delivery and transmits any data assuming as a byte stream. Hence, it fully utilizes the space in the command packets unlike XNP. The data object is represented as a set of fixed-size pages, so this method provides a manageable unit of transfer. This allows spatial multiplexing and supports efficient incremental upgrades.

MAC protocol: Regarding the MAC protocol, T-MAC is used as a MAC protocol and it increases the performances and prevents collisions. Deluge, on the other hand is using CSMA MAC, so it cannot prevent collisions.

Pipelining: Deluge supports pipelining in order to accelerate reprogramming in multi-hop networks. It allows large data transmission by fragmenting data into fixed size pages. It also supports pipelined page transmission to make dissemination faster. Unlike MOAP (Multihop Over-the-air Programming), nodes in Deluge should not wait for the complete code image before forwarding it.

Scope Selection Support: S2Torrent allows scope selection support, so only part of the node that is set to be reprogrammed could be selected. Deluge could disseminate one program to the whole network.

Idle listening to adaptive sleeping: All nodes need not to wait for the incoming packets all of the time like Deluge. As nodes wait for their requests to be fulfilled, a large amount of idle listening takes place and the radio needs to be on at all times, so that the nodes can listen to all of the messages.

Redundant to Request: Large data transmission is allowed by fragmenting data into fixed-size pages. It also supports pipelined page transmission to make dissemination faster. Unlike MOAP, nodes in S2Torrent shouldn't wait for the complete code image before forwarding it.

3. Simulation Parameters

Simulation length in the obtained simulation was 399.6998 seconds and the number of nodes that were included in the simulations was 200. Number of sending nodes is 35, and the same is the number of receiving nodes. There were 125,935 generated packets and 112,197 sent packets. Number of forwarded packets in the simulations is 64,769, and the number of dropped packets is 46,976. Number of lost packets in the simulations is 97,961. Minimum packet size that was used in the simulations is 32, while maximum packet size is 1078.

Hence, average packet size in the simulations is 877.2039, number of sent bytes is 103887174, and number of forwarded bytes is 56183524, while the number of dropped bytes is 39833944.

Other simulation parameters that were used in the simulations are:

- CPU active in current state – 1.8 mW;
- Radio current in receive state – 23 mW;
- Radio current in transmit state – 21 mW;
- Radio current in sleep state – 1 μW;
- EEPROM current in write state – 20 mW;
- EEPROM current in read state – 4 mW;
- EEPROM current in sleep state – 2 μW.

Furthermore, we will present the simulation parameters with the involving design factors. Table 1 presents the experimental setup and assumption.

Table 1. Experimental Setup and Assumption

Sr. No.	Parameters	Sample Value/Existing	Experimental Setup /Assumption (Proposed)
1.	Channel Type	Wireless Channel	Wireless Channel
2.	Wireless Propagation type		TwoRayGround
3.	Network Interface type		Wireless Phy.(T-MAC)
4.	No. of nodes in network		100
5.	Node Arrangement	Square Grid	Random
6.	Size of Topology in direction of X	1000 m	1800 m
7.	Size of Topology in direction of Y	500 m	900 m
8.	Protocol Type		Flood(Dissemination)
9.	Simulation Area	500m x 500m	1400m x 700m
10.	Interface Queue Type	Queue/Drop tail/PriQueue	Queue/Drop tail/PriQueue
11.	The maximum length in interface queue/ Maximum queue length	50	35
12.	Control Packet length	10 bytes	8 Bytes
13.	Data packet length	Up to 250 bytes	Up to 250 bytes
14.	MAC Type(MAC Layer protocol)		IEEE 802.11 (T-MAC)
15.	Mobility		No Mobility
16.	Object Size		31
17.	Density		400ft/m2
18.	Packet size	64 bytes	64 bytes
19.	Link layer type	LL	LL

Table 2 presents the MAC parameters that were set in the simulations. In Table 2 *byte_tx_time* presents the time needed to transmit a byte expressed in milliseconds. It is derived from bandwidth. Then, *slotTime* is used to denote the time of each slot in contention window. It should be large enough to receive the whole start symbol but cannot be smaller than clock resolution in msec. In this context, *slotTime_sec_* is used to express the slot time in seconds. The parameter *difs_* presents the DCF interframe space from 802.11 expressed in ms. It is used at the beginning of each contention window. It's the minimum time to wait to start a new transmission.

Then, *sifs_* presents short interframe space from 802.11 in ms. It is used before sending an CTS or ACK packet. It takes care of the processing delay of each packet. After that, *eifs_* in Table 2 presents the intended interframe space from 802.11 in ms. It is used for backing off in case of a collision. On the other hand, *guardTime_* is used as a guard time at the end of each listened interval in ms.

Table 2. MAC Parameters

Sr. No.	Parameters /Operation	Value (Proposed) TMAC
1.	byte_tx_time_	= 8.0 / BANDWIDTH
2.	start_symbol = byte_tx_time	* 2.5 // time to tx 20 bits
3.	slotTime_ = CLOCKRES >= start_symbol ? CLOCKRES	: start_symbol // in ms

4.	slotTime_sec_ = slotTime_	_ / 1.0e3 // in sec
5.	difs_	10.0 * slotTime_
6.	sifs_	5.0 * slotTime_
7.	eifs_	50.0 * slotTime_
8.	guardTime_	4.0 * slotTime_

Table 3 is presenting the AODV parameters that are used for the simulations and its values. It can be seen that the parameter *DISCOVERY_PERIOD* is set to 100 ms, and RETRY_INTERVAL is 30 ms.

Table 4 presents the Deluge parameters. We can notice that the value of the *DELUGE_VERSION* is 2.

Table 3. AODV Parameters

Sr. No.	Parameters	Value (Proposed)
1.	DISCOVERY_PERIOD	= 100 ms
2.	MAX_INTENTS_GLOBAL	= 2
3.	SEND_QUEUE_MAX_RETRIES	= 2
4.	DEFAULT_TTL	= 7
5.	AODV_MAX_METRIC	= DEFAULT_TTL
6.	SEND_QUEUE_SIZE	=1
7.	QUEUE_SIZE	=1
8.	AODVR_NUM_TRIES	=2
9.	AODV_RTABLE_SIZE	=7
10.	AODV_RQCACHE_SIZE	=7
11.	AODV_DATACACHE_SIZE	=7
12.	RETRY_INTERVAL	=30 ms

Table 4. Deluge Parameters

Sr. No.	Parameters	Value (Proposed)
1.	DELUGE_VERSION	2,
2.	DELUGE_MAX_ADV_PERIOD_LOG2	22,
3.	DELUGE_NUM_NEWDATA_ADVS_REQUIRED	2,
4.	DELUGE_NUM_MIN_ADV_PERIODS	2,
5.	DELUGE_MAX_NUM_REQ_TRIES	1,
6.	DELUGE_REBOOT_DELAY	4,
7.	DELUGE_FAILED_SEND_DELAY	16,
8.	DELUGE_MIN_DELAY	16,
9.	DELUGE_PKTS_PER_PAGE	48,
10.	DELUGE_PKT_PAYLOAD_SIZE	23,
11.	DELUGE_DATA_OFFSET	128,
12.	DELUGE_IDENT_SIZE	128,
13.	DELUGE_INVALID_ADDR	(0x7fffffffL),
14.	DELUGE_MAX_REQ_DELAY	(0x1L
15.	DELUGE_NACK_TIMEOUT	(DELUGE_MAX_REQ_DELAY
16.	DELUGE_BYTES_PER_PAGE	(DELUGE_PKTS_PER_PA

		GE*DELUGE_PKT_PAYLOAD_SIZE),
17.	DELUGE_PKT_BITVEC_SIZE	(((DELUGE_PKTS_PER_PAGE-1)/8 + 1)
18.	DELUGE_MAX_IMAGE_SIZE	(128L*1024L),
19.	DELUGE_MAX_PAGES	128,
20.	DELUGE_CRC_SIZE	sizeof(uint16_t),
21.	DELUGE_CRC_BLOCK_SIZE	DELUGE_MAX_PAGES* DELUGE_CRC_SIZE,
22.	DELUGE_GOLDEN_IMAGE_NUM	0x0,
23.	DELUGE_INVALID_VNUM	-1,
24.	DELUGE_INVALID_IMGNUM	0xff,
25.	DELUGE_INVALID_PKTNUM	0xff,
26.	DELUGE_INVALID_PGNUM	0xff,

4. Simulation Results

In this section the results that were obtained after detailed research simulations are presented. Table 5 gives results about the Overall Completion Time, expressed in seconds, against the object size (pages). In the obtained simulations Object Size includes 31 pages. Overall Completion Time is observed for the existing reprogramming protocol Deluge and the proposed reprogramming protocol with two approaches, S2TORRENT and S2TORRENT with Selective approach.

Results presented in Table 5 are also shown with graphs in Figure 1 in order the obtained values for the three reprogramming protocols to be better compared. As object page number increases, Overall Completion Time has tendency of increasing for all three reprogramming protocols. But, it is clearly shown in Figure 1 that Deluge protocol has largest values for the Overall Completion Time. S2 TORRENT protocol has lower values than Deluge, and the lowest values for Overall Completion Time has the proposed S2 TORRENT with Selective approach protocol. Maximum value of Overall Completion Time for Deluge protocol is 30.33 seconds, for S2TORRENT it is 24.41 seconds, and for S2TORRENT with Selective approach it is 23.69 seconds. We can also conclude that Overall Completion Time values are lowest for all object page numbers of the S2 TORRENT with Selective approach protocol.

Table 5. Object Size (Pages) against Overall Completion Time in seconds for Deluge, S2TORRENT and S2TORRENT with Selective approach

(Obj.) Page No.	Deluge (Sec)	S2TORRENT (Sec)	S2TORRENT with Selective approach (Sec)
1	3.653136531	1.49446494	0.22140221
2	5.756457565	2.98892989	1.10701107
3	7.638376384	3.98523985	1.16236162
4	8.856088561	5.64575646	2.3800738
5	10.68265683	7.25092251	2.49077491
6	12.39852399	8.35793358	4.09594096
7	14.72324723	9.79704797	4.15129151
8	16.66051661	11.5682657	6.64206642
9	18.59778598	12.5092251	6.69741697
10	20.03690037	13.1180812	8.85608856

11	21.69741697	14.00369	8.91143911
12	22.41697417	15.6088561	11.5682657
13	22.91512915	16.3284133	11.6236162
14	24.29889299	17.103321	13.8376384
15	25.2398524	17.7121771	13.8929889
16	25.79335793	18.8191882	14.8339483
17	25.95940959	19.095941	15.0553506
18	26.23616236	19.095941	16.5498155
19	27.50922509	19.3173432	16.4944649
20	27.78597786	19.3173432	16.4944649
21	28.06273063	19.095941	16.9372694
22	28.83763838	19.9261993	17.4354244
23	29.39114391	20.0369004	18.4317343
24	29.55719557	20.3136531	19.095941
25	29.55719557	20.8671587	19.8154982
26	29.72324723	21.9188192	20.5904059
27	29.88929889	23.0258303	21.6420664
28	30.05535055	23.7453875	22.3062731
29	30.22140221	24.1881919	23.0258303
30	30.33210332	24.1881919	23.4132841
31	30.33210332	24.4095941	23.6900369

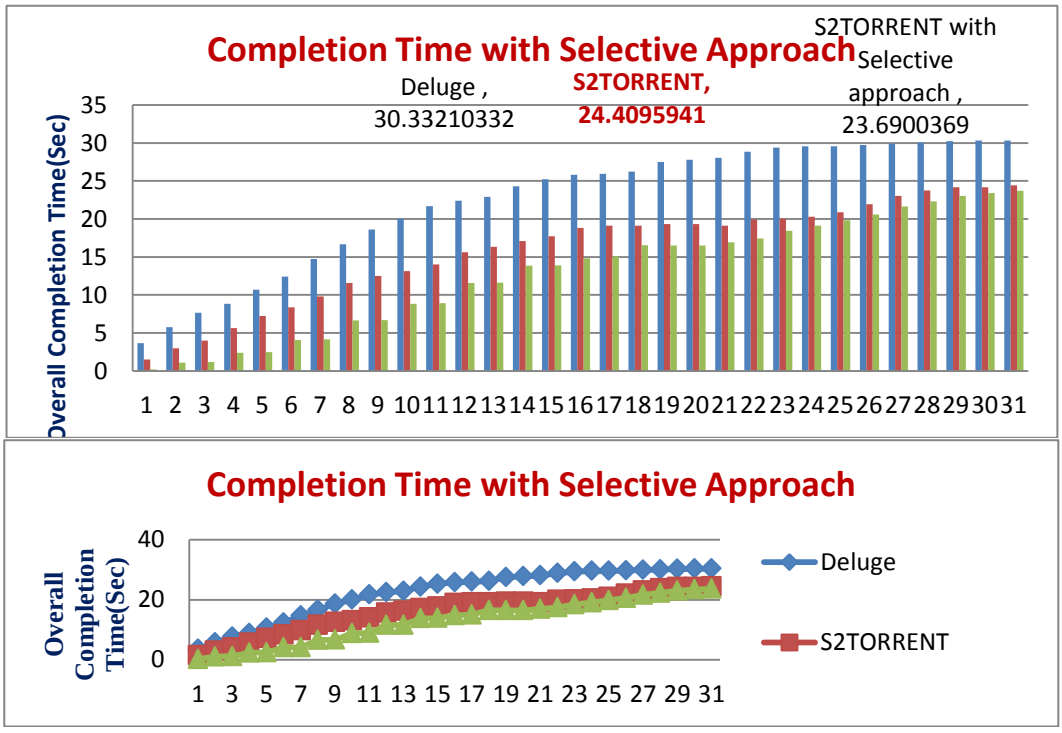


Figure 1. Overall Completion Time for Reprogramming Protocols

Figure 2 presents the values of the Average Overall Completion Time of the results presented in Figure 1. Deluge reprogramming protocol obtains average value of 22.41

seconds for Overall Completion Time. S2TORRENT reprogramming protocol gives lower average value of 15.96 seconds, and the proposed S2 TORRENT with Selective approach reprogramming protocol obtains the lowest value of 13.01 seconds for the Overall Completion Time.

From the perspective of the first test results for Deluge protocol (x1), second and third observation (x2) and (x3) for S2TORRENT and S2TORRENT with Selective approach protocol gives:

$$(x1-x2)/x1\cdot 100=(22.41-15.96)/22.41\cdot 100=28.78\% \quad (1)$$

$$(x1-x3)/x1\cdot 100=(22.41-13.01)/22.41\cdot 100=41.93\% \quad (2)$$

So, from equation (1), as well as from Figure 2 we can conclude that S2TORRENT requires 28.78% less completion time than the Deluge reprogramming protocol. On the other hand, from equation (2), and from Figure 2, we can see that S2TORRENT with Selective approach requires 41.93% less completion time than Deluge reprogramming protocol.

Figure 3 presents suppression results for the reprogramming protocols. In this case, 100 nodes are tested and results for each of them are presented in Figure 3. On the y-axis of Figure 3 advertisement rate is expressed in seconds for each of the 100 tested nodes.

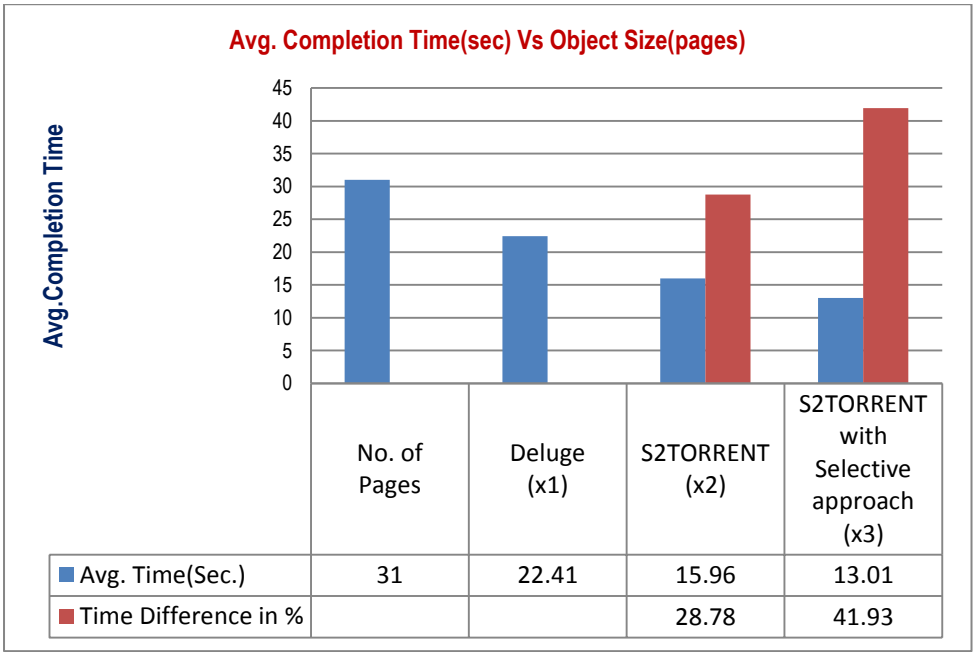
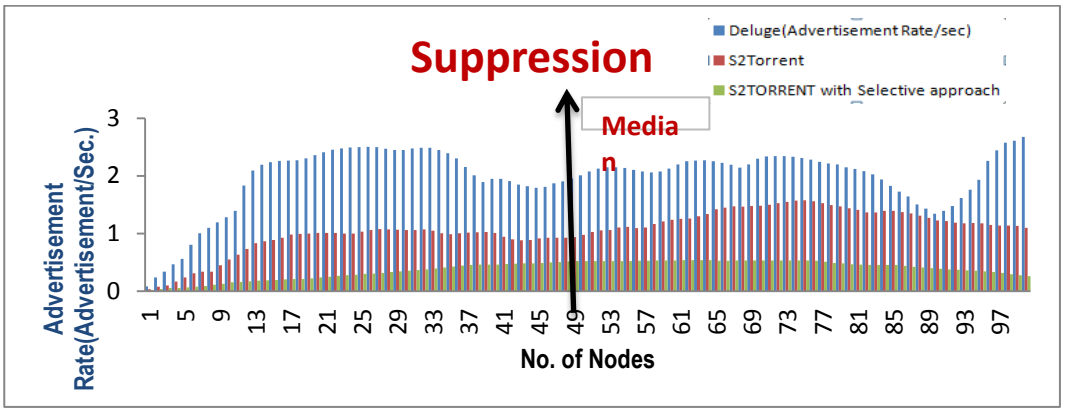


Figure 2. Average Overall Completion Time for Reprogramming Protocols



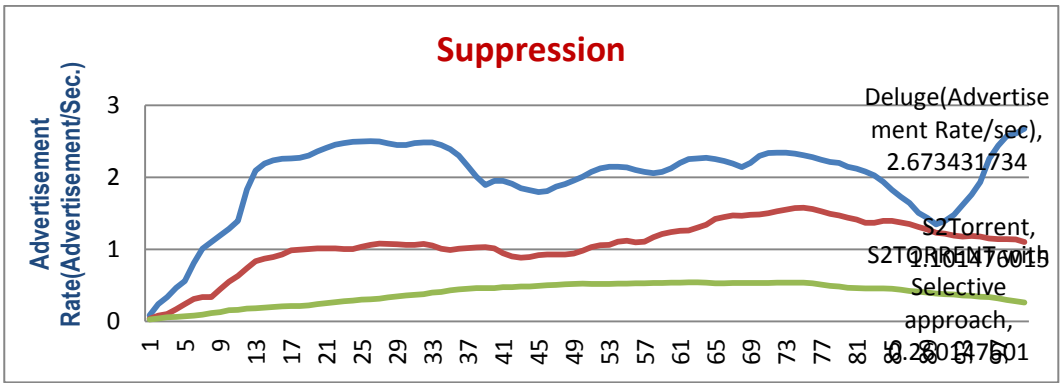


Figure 3. Suppression Results for Reprogramming Protocols

It is clearly shown in Figure 3 that S2 TORRENT with Selective approach protocol gives the lowest results for the measured advertisement rate.

Figure 4 presents the obtained results for transmitted request rate of 31 page object. Average numbers of transmitted requests per second against nodes in % are given in Figure 5.

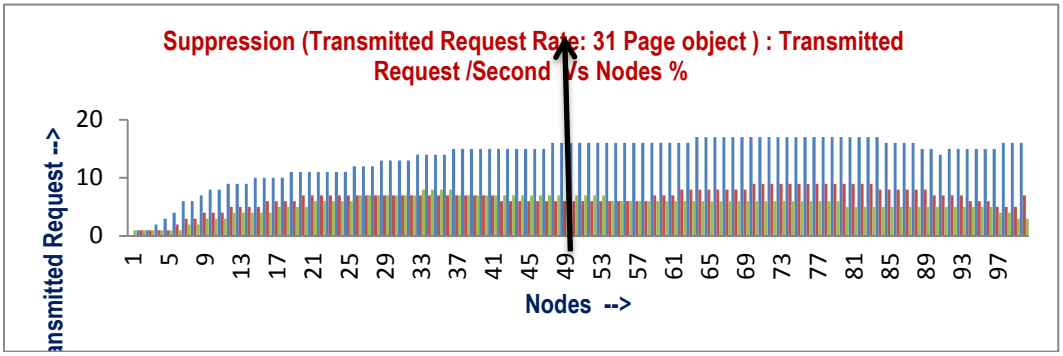


Figure 4. Suppression Results expressed in Transmitted Request Rate for Reprogramming Protocols

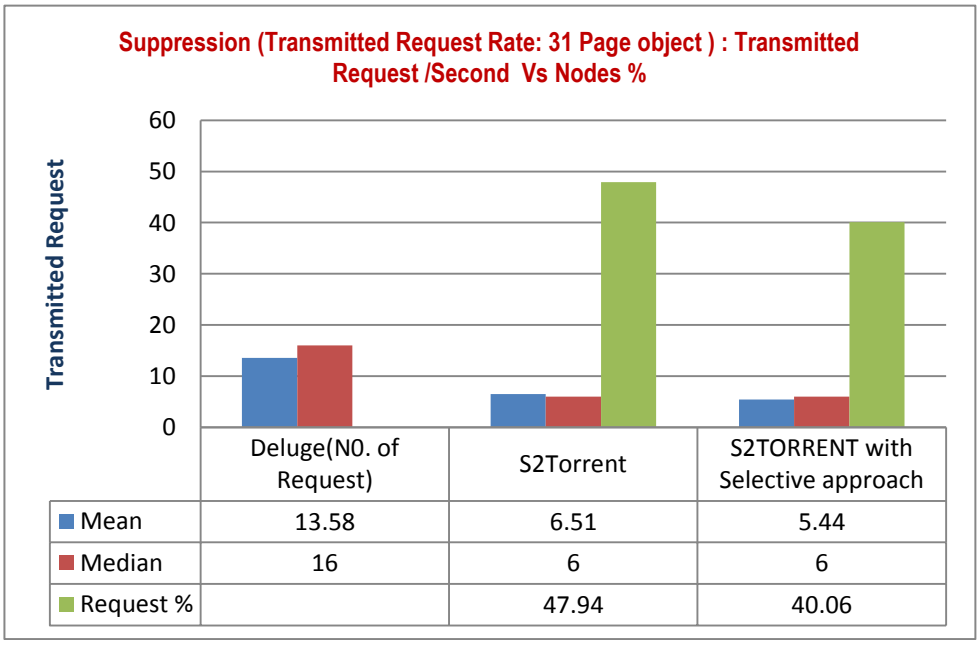


Figure 5. Average transmitted Requests for Reprogramming Protocols

We can conclude from Figure 5 that mean and median results of the transmitted requests are lowest for S2TORRENT with Selective approach protocol.

Figure 6 presents the Completion Time of each of the 100 nodes that were used in previously presented results. We can see that the Completion Time includes Init Time, Sign Time, Verify Time, Encrypt Time, Decrypt Time, and Key Establish Time. Each of these 6 types of times is expressed in milliseconds. We can conclude here that Completion Time of the nodes varies from 2.6 to 3.4 seconds.

Figure 7 presents energy consumption for each of the included times in the Completion Time. Energy consumption is expressed in mJ. We can see that Energy Consumption for the nodes is fluctuating between 800 and 900 mJ.

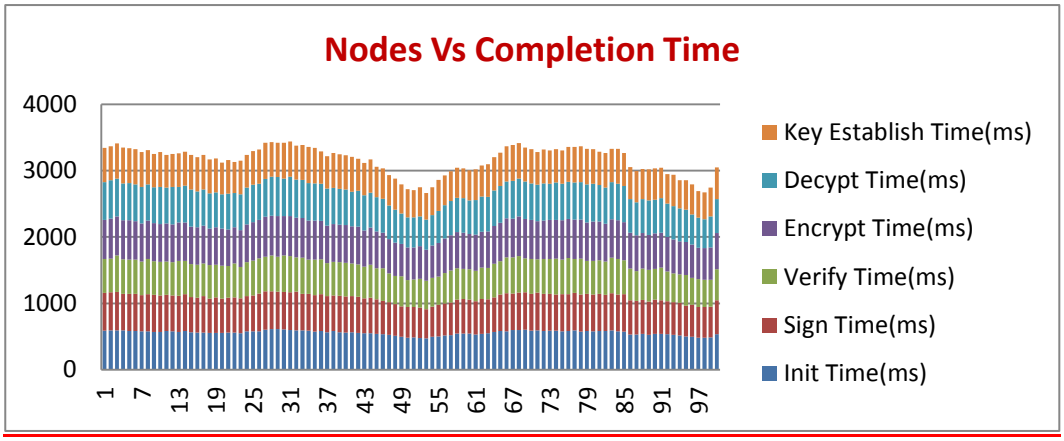


Figure 6. Completion Time of the Nodes

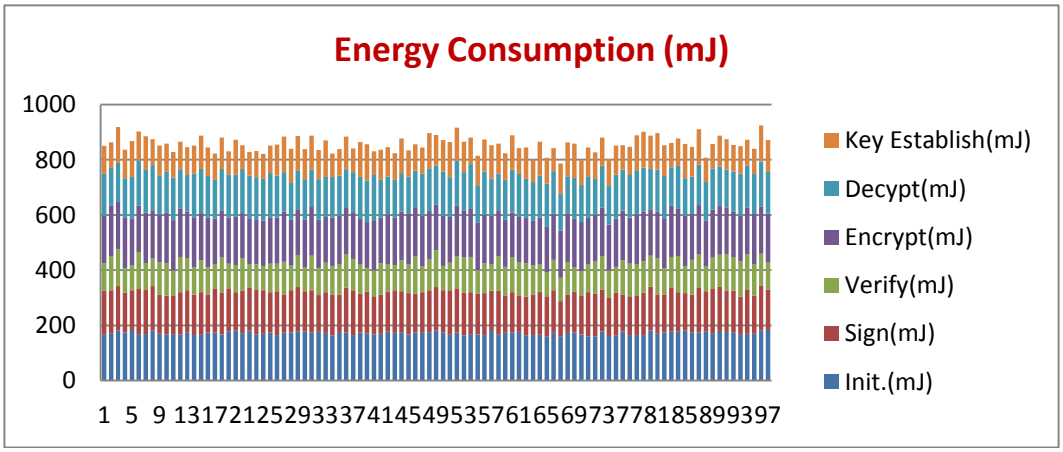


Figure 7. Energy Consumption of the Nodes

Conclusion

In this work we researched in details three reprogramming protocols that are very important for wireless sensor networks. We have obtained performance analysis of the main performances of Deluge, S2Torrent, and S2Torrent with Selective approach reprogramming protocols.

We can conclude that the proposed S2Torrent with Selective approach protocols has the lowest Overall Completion Time. Advertisement rate is also the lowest with this

protocol, if we compare it with the other two measured protocols. Hence, if we use the proposed S2Torrent with Selective approach protocol for reprogramming in wireless sensor networks, we will get the best performances from the networks topology.

References

[1] N. Patel, D. Culler and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks", Computer Science Division, University of California, **(2003)**, pp. 15-28.
[2] T. Arampatzis, J. Lygeros and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks", Intelligent Control, 2005, Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation, IEEE, **(2005)**, pp. 719-724.
[3] Q. Wang, Y. Zhu and L. Cheng, "Reprogramming wireless sensor networks: challenges and approaches", Network, IEEE, vol. 20, no. 3, **(2006)**, pp. 48-55.
[4] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo and D. Culler, "Tinyos: An operating system for sensor networks", Ambient intelligence, Springer Berlin Heidelberg, **(2005)**, pp. 115-148.
[5] D. Kumar, T. C. Aseri and R. B. Patel, "EEHC: Energy efficient heterogeneous clustered scheme for wireless sensor networks", Computer Communications, vol. 32, no. 4, **(2009)**, pp. 662-667.
[6] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of MAC protocols in wireless sensor networks: A survey", Communications Surveys & Tutorials, IEEE, vol. 15, no. 1, **(2013)**, pp. 101-120.
[7] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale", Proceedings of the 2nd international conference on Embedded networked sensor systems, ACM, **(2004)**, pp. 81-94.
[8] R. K. Panta, I. Khalil and S. Bagchi, "Stream: Low overhead wireless reprogramming for sensor networks", INFOCOM 2007, 26th IEEE International Conference on Computer Communications, IEEE, **(2007)**, pp. 928-936.
[9] M. Rossi, G. Zanca, L. Stabellini, R. Crepaldi, A. F. Harris III and M. Zorzi, "Synapse: A network reprogramming protocol for wireless sensor networks using fountain codes", Sensor, Mesh and Ad Hoc Communications and Networks, SECON'08, 5th Annual IEEE Communications Society Conference, **(2008)**, pp. 188-196.
[10] W. Dong, C. Chen, X. Liu, J. Bu and Y. Gao, "A lightweight and density-aware reprogramming protocol for wireless sensor networks", Mobile Computing, IEEE Transactions, vol. 10, no. 10, **(2011)**, pp. 1403-1415.

Authors

Kire Jakimoski, he received his B.Sc. degree in the field of Telecommunications from the Military Academy "Mihailo Apostolski" in Skopje, R. Macedonia in 2002, M.Sc. degree in Electrical Engineering in the field of Telecommunications from the Ss. Cyril and Methodius University in Skopje, R. Macedonia in 2007, and Ph.D. in technical sciences from the Ss. Cyril and Methodius University in Skopje, R. Macedonia in 2013. From 2002 to 2006 he works as an Officer for Telecommunications in the Ministry of Defense in the Republic of Macedonia. From January, 2006 to March, 2012 he works as an adviser for information security in the Directorate for Security of Classified Information in the Republic of Macedonia. From March, 2012 he is with the Faculty of Informatics, FON University in Skopje. Also, he is an author/co-author of around 40 published research papers and one book. He is an Assistant Professor and Vice Dean at the Faculty of Informatics, FON University in Skopje, Macedonia. His research interests include Wireless and Mobile Networks, Heterogeneous Wireless Networks, Computer Networks, Digital Telecommunications, Information Security.

Sime Arsenovski, he is a Full Professor at the Faculty of Informatics, FON University, Skopje, Republic of Macedonia.

Slavcho Chungurski, he is an Associate Professor and Dean at the Faculty of Informatics, FON University, Skopje, Republic of Macedonia.

Oliver Iliev, he is a Full Professor at the Faculty of Informatics, FON University, Skopje, Republic of Macedonia.

