ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse341942020.pdf
https://doi.org/10.30534/ijatcse/2020/341942020

Stream Control Transmission Protocol for IoT Data Communication Framework

[1]M. Padma, [2]Dr.N. Kasiviswanath
[1]Research Scholar, Department of Computer Science, Rayalaseema University, Kurnool.
[2]Professor and Head, Department of Computer Science & Engineering, GPREC, Kurnool.

ABSTRACT

In recent times, the Internet services were focused on providing the user with access to remote services such as access to read and store the files through web services and search. With the advent of access and service integration for various industrial and household electronic devices through online computing, cloud storage, the Internet of Things has become so pervasive. In this paradigm, several approaches are being added, which can communicate the sensed data through the Internet. For any IoT system to function with an agreeable performance it must have to be in tune with the proper semantics be defined for the cross-layer communication, adequate discovery for the services as well as implementing the context-awareness through the middleware layers, and data be distributed across multiple server locations to balance the load. For all these vital features to be implemented in the IoT system, with the implementation of TCP it is observed that there is a considerable degradation at quality of service (QoS) because TCP needs more messages to be transacted for every transaction, TCP is vulnerable to frequent Timeouts which results to constant computation of Retransmission Timer Timeout. Hence, there is a need for a protocol which would be a better alternative to TCP, which further should facilitate the system with streaming the data and having lesser messages to be exchanged for every transaction. This paper enlightens the IoT Data communication performance using SCTP protocol rather than TCP among functional, on-functional and architectural environment at two different time instances of transmission parameters such as End-to-End Delay and Throughput.

Key words: Internet of Things, Cross-layer Model, Context-aware service discovery, Distributed data service, SCTP (Stream Control Transmission Protocol).

1.INTRODUCTION

Internet of Things (IoT) is an evolution of the current Internet into a ubiquitous network of interconnected objects communicating and sharing information with each other, which enables them to see, hear, think and take actions. It is enabled by embedding identification, sensing, communication and actuation capabilities into the devices used on a daily basis [1, 2].

The 'things' in IoT comes with a wide variety of hardware specifications, communication capabilities and Quality of Service (QoS) requirements making it highly heterogeneous. The devices differ primarily in computation capabilities, power and memory specifications, supporting communication mechanisms, energy consumption levels, delay and reliability measurements [1, 2, 3].

This paper enlightens the IoT Data communication performance using SCTP protocol among functional, non-functional and architectural environment at two instances of Transmission parameters such as End-to-End Delay and Throughput.

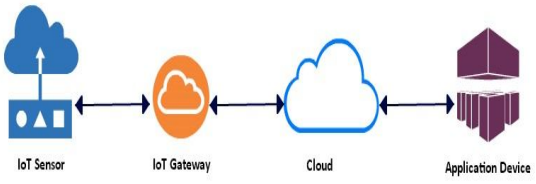


Figure 1: Typical IoT System Scenario

Scenario -1: Cross-layer framework, which considers as interoperable services from the IoT Architectural requirements,the cross-layer design is normally of the essence in addressing connectivity issues among the devices in the Internet of Things (IoT). The cross-layer design as a model helps in getting rid of strict boundaries of the OSI networking model, thereby allowing data access of another layer.

There are different layers in data sharing in the Internet of Things (IoT), such that, sharing of data across the different segments becomes a challenge. With the use of the cross-layer design model, different layers can exchange information with each other, such that they practically interact more towards improving data communication [4].

Scenario -2:IoT Service discovery under a context-aware model which deliberate from IoT architectural requirement, Middleware layers are intended to anticipate the context from the client processes to deliver the appropriate services [5]. Whereas, poor context-awareness, both at the client as well as the server middleware, results in an inappropriate request from the client, which would further lead to inadequate services from the server middleware. In this context, current IoT systems suffer from insufficient context awareness of services due to inexpertly modelled semantics proliferating various unevenly distributed ontologies and incoherent semantics for services. Similarly, from an M2M viewpoint, service discovery is still unconvincing. Eventually, it can be stated that Poor Information Visualization and Analysis.

Scenario-3:In IoT DDS is an open standard for a data-centric publish-subscribe middleware platform[7] with a rich set of real-time and QoS capabilities published by the OMG (Object Management Group)[6].In a distributed system, middleware is the software layer that lies between the operating system and applications. It enables the various components of a system to communicate and share data more efficiently. It simplifies the development of distributed systems by letting the software developer's focus on the specific purpose of their applications rather than the mechanics of passing information between applications and systems. In DDS-IoT platform that facilitates device inter-communication, logic and applications at every layer of the Internet of Things infrastructure.

From the above scenarios, we measured the IoT data communication performance using the SCTP protocol, and it concludes the significance of SCTP protocol[8] in different scenarios towards pointing Data communication in IoT framework.

Rest of the paper is organized as Section II describes the problem definition,Section III describes System Study, which explains three scenarios. Section IV describes result and analysis; finally, Section V presents the conclusion.

2.PROBLEM DEFINITION

This section describes the insights around the limitations of the current IoT functional, Non-functional and Architectural environment that are preventing the deployment of the new IoT scenario. Some of the limitations include the dynamic and ultra-large-scale nature of the IoT infrastructure, which invalidates centralized resource registries and discovery approaches. Lack of standardized description of services has posed a significant challenge in the issue while integrating the various services. In Cross-layer Communication, the new network has come along with numerous challenges, such as security, privacy, efficiency as well as reliability apart from other problems. In the Cross-layer Communication model, different layers can exchange information with each other [9], such that they practically interact more which improves the communication performance in terms of the minimizing the communication overhead for every message transaction.

Middleware layers are intended to anticipate the context from the client processes so as to deliver the appropriate services. Whereas, poor context-awareness, both at the client as well as the server middleware, results in an inappropriate request from the client, which would further lead to inadequate services from the server middleware. In this context, current IoT systems suffer from insufficient context awareness of services due to inexpertly modelled semantics proliferating various unevenly distributed ontologies and incoherent semantics for services.

To address the data storage efficiency a Distributed Data Service[10] for IoT Middleware would be a great advantage which shall be aware of minimum data storage resources in the IoT environment, as to how data would be collected and aggregated. Handling large amounts of data is the fundamental hazard in the IoT system, which limits and affects system performance.

From the above statement to improve the QoS[9] concerning IoT data communications, we first-rate the Transport layer protocol called SCTP (Stream Control Transmission Protocol), which provides the better commutations towards improving the QoS.

3.SYSTEM STUDY

Fundamentally, the data from the Internet of Things would be communicated to the IoT cloud through the cloud gateway. The cloud gateway further communicates the data to each subsystem as per the definitions in the request to satisfy the requirement. In the subsequent sections, each subsystem is substantiated with the relevant simulation architecture and simulation results. However, in each of the IoT subsystem, the specific functional modules are mentioned as soft components which are intended to implement a set of functions which comprehensively satisfy the requirements.

Most of the works have addressed the performance of the communication concerning the implementation using Transmission Control Protocol (TCP). Our work attempts to improve communication performance by using Stream Control Transfer Protocol (SCTP). The reason to adapt SCTP as the transport layer protocol is illustrated in table 1. Although the services such as connection type, data transfer reliability are standard in TCP and SCTP, but they differ in others, notably TCP does not support, such as message-based transfer, multihoming, multistreaming, error tolerance, protection from spoofed attacks during connection establishment, denying half-closed connections, reconfiguring the addresses dynamically, and preserving the message boundaries. Moreover, IoT data is sensitive, the data analytics are improved to handle any formats of data such as messages, and IoT systems anticipate to send message-based transfer instead of discrete messages or byte-oriented transmission. This study helps our work to conclude the selection of SCTP as the transfer layer protocol.

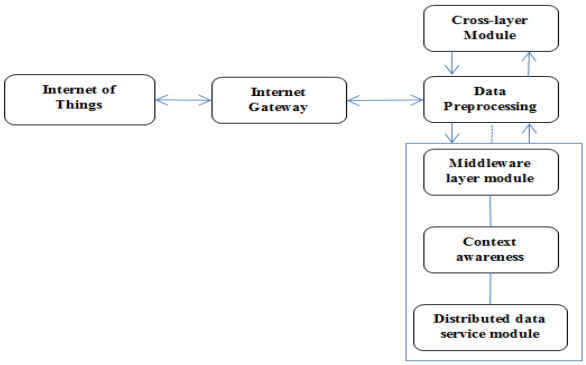


Figure 2: IoT Cross-Layer Design Approach

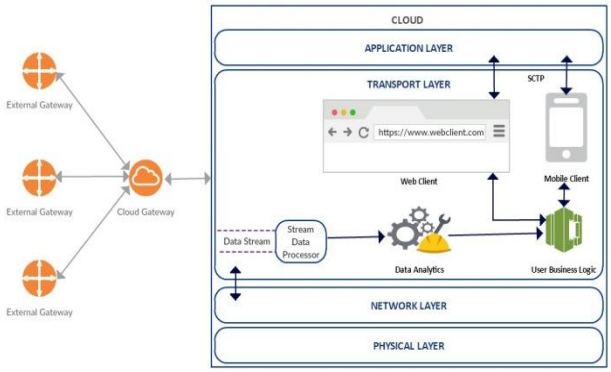


Figure 3: IoT Cross-layer Design Architecture

In the above architecture, the advantage of implementing a cross-layer approach is mentioned implicitly. Furthermore, the soft components, Stream Data Processor (SDP), Data Analytics (DAn), User Business Logic (UBL), Web Client (WB), and Mobile Client (MC) would play their defined roles in achieving the specified objective. External gateways communicate the IoT data that they have collected from various IoT devices (things) to the cloud gateway. Cloud gateway, in turn, sends this data to the SDP by using Stream Control Transfer Protocol (SCTP) wherein SDP acts as a data collection component. The collected data would be shared with DAn which analyses the data to match the UBL format. UBL processes the data as per the requirements of WC and MC. The subsystem takes advantage of using SCTP with the increased efficiency in communication by reducing the time it takes to communicate the request or the response. IoT systems cannot afford delay due to its property to attend sensitive decision within, almost, no time. This parameter of the shortest delay in communication is the primary requirement in important IoT systems such as healthcare IoT and aviation.

Service Discovery through Context Awareness

The proposal in this paper is based on the general architecture depicted in Figure 4.that consist of middleware platforms, end-users, and applications to retrieve IoT available resources such as (i.e., sensors, actuators, services, context data).

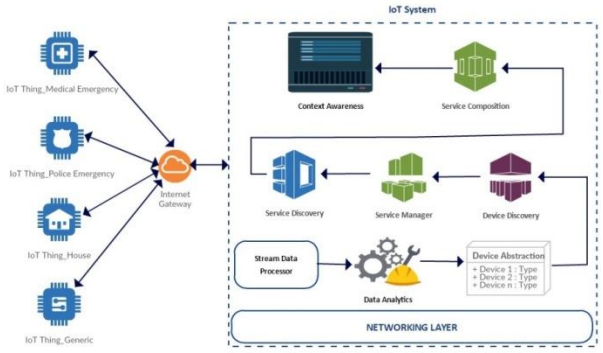


Figure 4: Proposed Architecture

The architecture, as Figure 4.described in the diagram, proposes the placement of the solution for the middleware layer. Data is collected from various IoT devices through the Internet Gateway, which is further processed by the Stream Data Processor (SDP). SDP is responsible for ensuring fault tolerance among the data streams and computational scales. The Data Analytics (DAn) receives the acquired data from SDP to interpret the data and analyse the results using statistical techniques. Device Abstraction supports the Device Discovery process to detect the attributes of the devices automatically to invoke the relevant functions of requesting for the services. Service Manager (SM) checks the service level agreements (SLAs) with external service providers and customers which further assists Service Discovery (SD) module to discover the appropriate service and invoke the service composition module (SC) to process the identified services, store them. These services from SC would be used by Context-Awareness (CA) module to analyse and derive a suitable context for the use of a requesting IoT device.

Distributed Data Service Architecture for IoT

The proposed architecture for the IoT middleware addresses the problem to provide the solution in the way the streamed data is communicated to the respective modules. The architecture is prepared by including the key components to make the communication of IoT data be streamlined in order to produce faster communication. Soft components such as Stream Data Processor (SDP), Data Analytics (Dan), Device Abstraction (DAb), Device Discovery (DD), Resource Manager (RM), Resource Discovery (RD), Service Composition (SC), and Context Awareness (CA) are being the part of the Application layer, interact as per the architectural definition, as shown in the Figure 5.

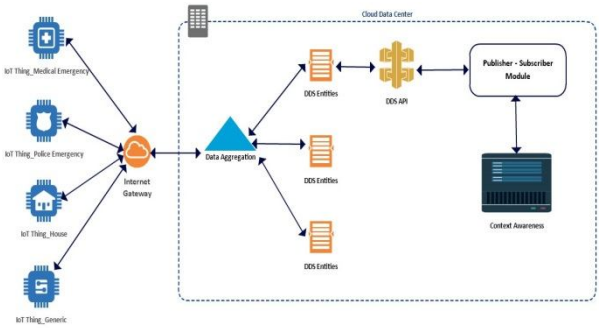


Figure 5: Proposed Distributed Data Systems Architecture for IoT

The real-time data from the various Internet of Things are sent to the cloud gateway, which further transfers the data to the processing module through the SDP. SDP accommodates the data to be received by it as per the format that the next soft component, DAn, can process. DAn processes the data according to the requests that are accepted for a service to be offered. Meanwhile, the other soft components, DD identifies the device to which the service to be provided through the DAb; through RM, RD identifies the resources to be offered to SC. All the data is aggregated to finalize the context by which the DAn further decides the action to be initiated. However, the SC keeps receiving the relevant data through the Networking layers through SCTP.

Table.1: Service comparison between TCP and SCTP

Services	TCP	SCTP
Connection-oriented	Yes	Yes
Reliable Data Transfer	Yes	Yes
Message-based Transfer	No	Yes
Multihoming	No	Yes
Multistreaming	No	Yes
Error Tolerance	No	Yes
Protection from spoofed SYN attacks	No	Yes
Allow half-closed connections	Yes	No
Dynamic address reconfiguration	No	Yes
Preserve message boundaries	No	Yes

4.RESULT AND ANALYSIS

The results directory contains .vec and .sca files, which are the files that store the results in vector and scalar form, respectively. Vectors record data values as a function of time, while scalars typically record aggregate values at the end of the simulation.

Our Simulation results captured from OMNET 4.2 ++ Simulation tool, Operation System Win 7/8/10, Code has been developed in C++. From these results, we measure the data broadcasting performance in IoT Cross-layer Communication, IoT Service discovery through context-aware and IoT Distributed Data Service using SCTP protocol. In this connection, we calculate the performance measuring constraints by datagram count, Mean, StdDev and Variance in the above scenarios.

4.1 Calculating Network Throughput on Layered Networks

Throughput in Packet-Based Networks

Calculating throughput is very straightforward. We have to calculate how much of something is handled in a second. In the case of networking, this boils down to how much data is transmitted or received in a second:

$$Throughput = data / time\ [s]$$

We should define what traffic is part of the data variable in the throughput calculation. This depends on the task at hand:

- All outgoing data on an interface.
- All incoming data on an interface.
- All incoming data on an interface destined for that interface.
- All incoming or outgoing data within a data flow.
- All incoming data are originating from a specific source.

Furthermore, within the context of packet-based networks, the relevant data may be counted or measured in two different units.

- number of packets
- number of bits

The resulting throughput will thus be measured in $packets\ per\ seconds$ ( )PPS or bits per $second$ (bps).These two measurements are closely related.

$$Throughput\_packets\ [pps] = data\ [frames] / time\ [s]$$

$$Throughput\_bits\ [bps] = data\ [bits] / time\ [s]$$

$$bit\ rate\ [bps] = Throughput\_packets\ [pps] * packet\_size\ [bits/packet]$$

4.2 Comparison of Quality of Service Parameters

4.2.1 Cross-layer: End-to-End Delay with SCTP

a. End-to-End Delay with SCTP

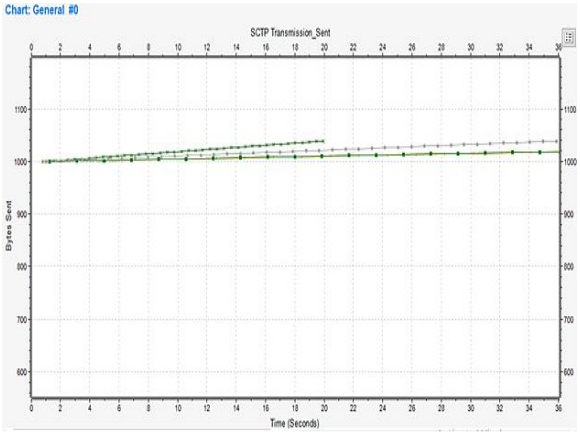


Figure 6: End-to-End Delay with SCTP

Table 2.: Instance of Transmission End-to-End Delay with SCTP

Instance	No. of Bytes	Transmission Start Time (in Seconds)	Delay (in seconds)
1	1000	0.799	0.0025858
2	1000	16.07	0.0025323

Improvement in End-to-End Delay = 0.0025858 - 0.0025323 = 0.0000535 seconds.

In the simulation of cross-layer design, first instance of the transmission from one end of SCTP to the other end of SCTP starts at 0.799 seconds which consumed an end-to-end delay of 0.0025858 seconds whereas the second instance of communication started at 16.07 seconds overwhelmed an end-to-end delay of 0.0025323 seconds which has led to an improvement of 0.0000535 seconds, otherwise by 2.07%.

b. Throughput

Table 3: Instance of Transmission Throughput in SCTP

Instance	Transmission Start Time (in Seconds)	No. of Bytes Sent	No. of Bytes Received
1	1.52	1000	978
2	20.1	1037	1018

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

= (978 / 1000) * 100 = 97.8%

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

= (1018 / 1037) * 100 = 98.17%

To compute the throughput in simulation for the cross-layer design first instance of the communication was initiated at 1.52 seconds with the data size of 1000 bytes whereas the received data size was 978 bytes. In the second instance the communication was initiated at 20.1 seconds with data size of 1037 bytes whereas the received data size was 1018 bytes. As a function of the increase in time and an increase in data size, it is observed that the delivery of data is increased by 0.37%.

4.2.2 IoT Service Discovery through Context-aware

a. End-to-End delay

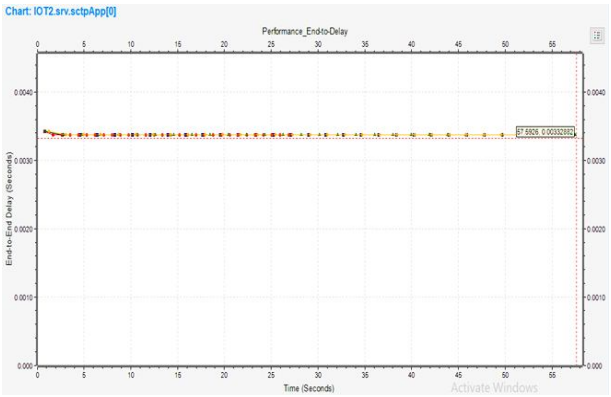


Figure 7:. End-to-End Delay

Table 4: Instance of Transmission End-to-End Delay with SCTP

Instance	No. of Bytes	Transmission Start Time (in Seconds)	Delay (in seconds)
1	1000	0.575926	0.00343
2	1000	0.7919	0.0033288

Improvement in End-to-End Delay = 0.0034300 – 0.0033288 = 0.0001012 seconds

In the simulation of middleware layer for service discovery the first instance of the transmission from one end of SCTP to the other end of SCTP starts at 0.575926 seconds which consumed an end-to-end delay of 0.00343 seconds whereas the second instance of communication started at 0.7919 seconds consumed an end-to-end delay of 0.0033288 seconds which has led to an improvement of 0.0001012 seconds, otherwise by 2.95%.

b. Throughput

Table 5: Instance of Transmission Throughput in SCTP

Instance	Transmission Start Time (in Seconds)	No. of Bytes Sent	No. of Bytes Received
1	0.35	995	971
2	26	1029	1012

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

= (971 / 995) * 100 = 97.59%

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

= (1012 / 1029) * 100 = 98.35%

To compute the throughput in simulation for the middleware layer first instance of the communication was initiated at 0.35 seconds with the data size of 995 bytes, whereas the received data size was 971 bytes. In the second instance, the communication was initiated at 26 seconds with a data size of 1029 bytes, whereas the received data size was 1012 bytes. As a function of the increase in time and an increase in data size, it is observed that the delivery of data is increased by 0.76%.

4.2.3 Distributed Data Service:

a. End-to-End Delay

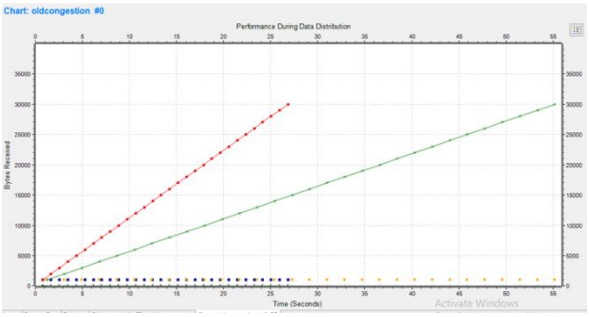


Figure 8:. End-to-End Delay

Table 6: Instance of Transmission End-to-End Delay with SCTP]

No. of Bytes	Transmission Start Time (in Seconds)	Delay (in seconds)
1000	35	1.8057
1000	143	1.303

Improvement in End-to-End Delay = 1.8057 – 1.303 = 0.5027 seconds

In the simulation of Distributed Data Services for an assistance to load balancing, the first instance of the transmission from one end of SCTP to the other end of SCTP starts at 35th second which consumed an end-to-end delay of 1.8057 seconds whereas the second instance of transmission started at 143rd second consumed an end-to-end delay of 1.303 seconds which has led to an improvement of 0.5027 seconds, otherwise by 27.84%.

Throughput in SCTP

Table 7: Instance of Transmission Throughput in SCTP

Instance	Transmission Start Time (in Seconds)	No. of Bytes Sent	No. of Bytes Received
1	0.798	1000	974
2	26.857	30000	29580

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

= (974 / 1000) * 100 = 97.4%

Throughput = (Total number of bytes received / Total number of bytes sent) * 100

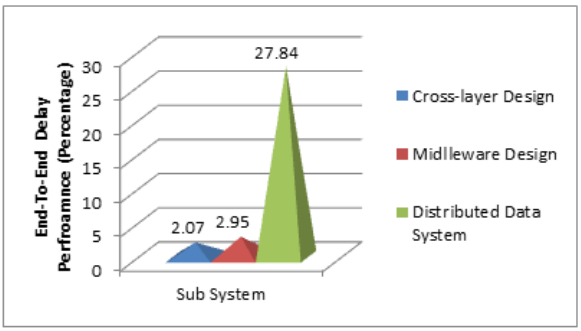
= (29580 / 30000) * 100 = 98.6%

To compute the throughput in simulation for the distributed data services, the first instance of the communication was initiated at 0.798 seconds with the data size of 1000 bytes, whereas the received data size was 974 bytes. In the second instance, the communication was initiated at 26.857 seconds with data size of 30000 bytes, whereas the received data size was 29580 bytes. As a function of the increase in time and an increase in data size, it is observed that the delivery of data is increased by 1.2%.

SCTP performance among three scenarios [End – to-End Delay] and [Throughput].

Table 8: .End-to-End delay [Over all Comparison among scenarios]

Subsystem	Transmission Start time Instance 1 (in seconds)	Transmission Start time Instance 2 (in seconds)	Improvement in end-to-end delay (in seconds)	Improvement in end-to-end delay (percentage)
Cross-layer Design	0.799	16.07	0.0000535	2.07%
Middle layer Design	0.575926	0.7919	0.0001012	2.95%
Distributed Data System	35	143	0.5027	27.84%

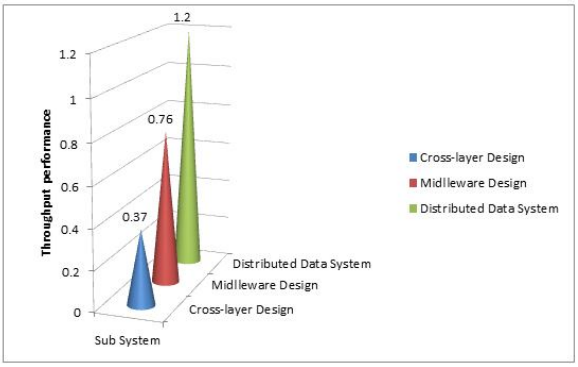


Graph: 1: End-to End delay performance

To compute the end-to-end delay in simulation for the above subsystems.The Instance of the transmission from one end of SCTP to the other end of SCTP. When the time at which the communication was initiated keeps increasing the performance of the end-to-end delay is increasing considerably, which means that as a function of time the performance of SCTP end-to-end delay keeps increasing

Table 9:.Throughput: [Over all Comparison among three scenarios]

Subsystem	Transmission Start time Instance 1 (in seconds)	Transmission Start time Instance 2 (in seconds)	delivery of data is increased (percentage)
Cross-layer Design	1.52	20.1	0.37%
Middle layer Design	0.35	26	0.76%
Distributed Data System	0.798	26.857	1.2%



Graph 2:. Throughput performance

To compute the throughput in simulation for the above subsystems, it seems that when data size increase, data delivery rate also increased.

5.CONCLUSION

The results that are observed through the simulations for all the three subsystems to compute the End-to-end delay and Throughput parameter, and when all the three results are compared, an interesting fact was observed. Consolidated results are as shown in the table. When the time at which the transmission was initiated keeps increasing the performance of the end-to-end delay is increasing considerably, which means that as a function of time the performance of SCTP end-to-end delay keeps increasing similarly delivery of data is increased when the throughput of data size increased. Hence we prove that SCTP is the best performer in IoT Data Communications in different context of communications.

REFERENCES

[1]. Chong Han, Josep Miquel Jornet, Etimad Fadel and Ian F. Akyildiz, "A cross-layer communication module for the Internet of Things," found in http://www2.ece.gatech.edu/research/labs/bwn/papers/ 2013/j6.pdf, accessed on 4th April 2016.

[2]. Ala Al-Fuqaha, Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications", IEEE Communication Surveys & Tutorials, Vol. 17, No. 4, Fourth Quarter 2015.

[3]. A. J. Stankovic,"Research Directions for the internet of things", IEEE Internet of Things Journal, vol. 1, no. 1, pp. 3–9, 2014.

[4] C. Luo, F. Richard Yu, H. Ji, and V.C.M. Leung, "Cross-layer Design for TCP Performance Improvement in Cognitive Radio Networks," IEEE Trans. Vehicular Technology, vol. 59, no. 5, pp. 2485-2495, June 2010.
https://doi.org/10.1109/TVT.2010.2041802

[5] Razzaque M.A., Milojevic-Jevric M., Palade A., Clarke S. Middleware for the Internet of Things: A Survey. IEEE Internet Things J. 2016;3:70–95. DOI: 10.1109/JIOT.2015.2498900.

[6] OMG, "Data Distribution Service for Real-time Systems," 2015.

[7] Fersi G. Middleware for Internet of Things: A Study; Proceedings of the International Conference on Distributed Computing in Sensor Systems 2015; Fortaleza, Brazil. 10–12 June 2015; pp. 230–235.

[8] Paul Stalvig (2014): Introduction to the Stream Control Transmission Protocol (SCTP): The next generation of the Transmission Control Protocol (TCP). Technical Report.Available at https://f5.com/resources/ white-papers/introduction-to-the-stream-control-transmission-protocol.

[9] Hasan, Mohammed Zaki, Fadi Al-Turjman, and Hussain Al-Rizzo. "Analysis of Cross-Layer Design of Quality-ofService Forward Geographic Wireless Sensor Network Routing Strategies in Green Internet of Things." IEEE Access 6, 20371-20389. 2018.
https://doi.org/10.1109/ACCESS.2018.2822551

[10] Cruz Huacarpuma, Ruben et al. "Distributed Data Service for Data Management in Internet of Things Middleware." Sensors (Basel, Switzerland) vol. 17,5 977. 27 Apr. 2017, doi:10.3390/s17050977