A Research on Interoperability Issues in Internet of Things at Application Layer

V. Tirupathi, K. Sagar

ABSTRACT---Now a days everyone is depending on smart things (smart computing devices) or devices to complete their task comfortably from anywhere. These smart things create a network to exchange information between them by following OSI/ISO (Open system interconnection) or TCP/IP protocol stack. The two protocol stacks have seven layers which are being used by smart things to communicate with each other through network. These seven layers perform different tasks by using different protocols at each level. For example, network layer has TCP and UDP protocols for segmentations and communications application layer has http, https and SNMP etc... Smart things communicate with each other either through connection or connection less communication mediums. Devices are various kind, some of them are operate on low-power and low-data rates. Some of them operate on high-power and high-data rates using strong Ethernet networks. Establishing communication among heterogeneous devices and software protocols is a challenging task. Smart things behave intelligently with help of internet, embedded sensors and actuators. Smart things have processing and network capabilities. These are the building blocks for Internet of Things (IoT). This paper summarizes all application layer protocols and interoperability issues.

Keywords—OSI/ISO, TCP/IP, UDP, IoT, SNMP

1. INTRODUCTION

Internet of Things (IoT) is "internet connectivity among physical devices and everyday objects". It has introduced by Kevin Ashton of Procter & Gamble in 1999. The things may be radio-frequency identification (RFID) tags, actuators, mobile phones, sensors and other devices which are capable of connecting with internet.

Internet of things (IoT) has expanded tremendously due to constant innovations in hardware, communication networks and software solutions [1, 2]. Huge number of devices are communicating with each other through internet and generating massive collections of data. To store and process huge collections of data cloud servers are being used [3, 4]. For fast processing of data and to get faster response fog computing technologies were introduced. The performance of IoT together with fog and cloud determined using many factors. Application layer protocols plays a vital role among these factors. HTTP and HTTPS protocols are extensively used protocols to make communication with other servers through internet. HTTP appropriate for computing devices with high-power, faster processing unit and strong communication mediums. But IoT devices have slower processing unit and low-power.

Few light weight protocols have created for IoT devices at application layer level. They are AMQP (Advance Message Queuing Protocol) and DDS (Data Distribution Services), CoAP (Constrained Application Protocol), MQTT (Message Queuing Telemetry Transport), XMPP (Extensible Message and Presence Protocol). These protocols support constrained devices for message exchange. In fact a single messaging protocol is not enough to provide all communication services because protocols uses different communication models.

2. COMMUNICATION MODELS

The IoT devices send data to and receive from servers or cloud systems using any one of the application layer protocols like CoAP or MQTT. These protocols either follow request/response or publish/subscribe communication model. Request/response communication widely used in client/server applications. Client sends request to the server, server receives client request process it and sends response to the clients. A server can process any number client requests at the same time. CoAP and HTTP follows request/response communication model. Fig1 shows request/response model.

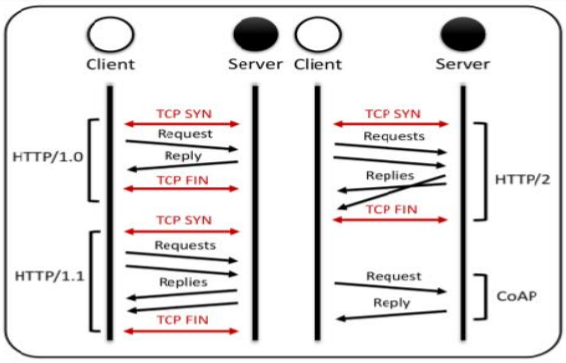

Fig 1: Request-Response model, for example: COAP and HTTP

Publish/subscribe is an alternative to request/response model. It consisting of three entities i.e. publisher, broker and subscriber. Fig 2 describe publish/subscribe model. Subscriber subscribes the required topic within the system. Broker is an important entity who manages incoming and outgoing messages in between publisher and subscriber. Publisher is a information provider [5] who provides services and send to the broker. MQTT, AMQP and DDS follows publish/subscribe communication model.

Revised Manuscript Received on June 10, 2019.

V. Tirupathi, Assistant. Prof Dept of CSE S R Engineering College Warangal, India, Thirupathi, A.P ,India.. (vadluri@srecwarangal.ac.in)

Dr K. Sagar, Professor Dept of CSE CBIT Hyderabad, Telangana, India (kadapasagar@cbit.ac.in)

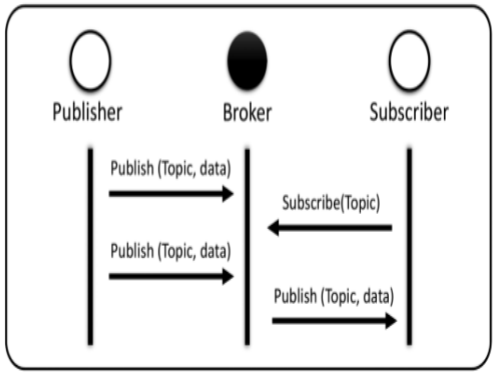

Fig 2: Publish-Subscribe model, for example: MQTT, DDS, and AMQP.

3. OVERVIEW OF APPLICATION LAYER PROTOCOLS

This section explains the application layer protocols by describing communication model, QoS, security features and transport protocols. CoAP runs on UDP (User datagram protocol), while HTTP, MQTT, AMQP, XMPP and HTTP run on TCP (Transmission control protocol) at transport layer level. As we know MQTT, DDS and AMQP follows publish/subscribe communication model. CoAP and HTTP follows request/response communication model.

3.1 HTTP

HTTP (Hyper Text Transfer Protocol) is basic client/server model intended for web applications by following request/response model. Client sends a request to the server, server process the request and sends response the client. Server can handle any number of client requests at given time. To implement web services recently HTTP associated with REST [11].

Fig 3: REST HTTP interaction model.

HTTP uses GET, POST, PUT and DELETE methods to exchange information between client and server. POST is securable method compare to GET method. TCP is being used as transport protocol which uses TCP three way handshake protocol to establish communication between client and server. It efficiently manages missed packages. HTTP parses data either in text format or JSON (Java Script Object Notation) format while exchanging data. HTTP follows TLS [12] security mechanism. HTTPS protocol has introduced to provide secure communications.

3.2 CoAP

CoAP (Constrained Application Protocol) invented to support constrained devices (IoT devices) by Constrained Restful Environments (CoRE) working group of IETF [7]. It follows request/response model as HTTP by using UDP at transport layer level to provide quality services faster. CoAP supports GET, POST, PUT and DELETE methods to get required resources by specifying the respective URI (Universal Resource Identifier) from the server. CoAP has two layers, first layer is for exchanging information between client and server by using the above methods with the help of UDP. But UDP does not provide reliable communications. CoAP's second layer being used to provide reliable data transfers by resending the lost packets. To ensure reliable packet transfer it uses four different types of messages they are CON (Confirmable), NON (non-confirmable), RST (reset), ACK (Acknowledgement). The CON message guarantee message transfers. If the receiver demands for acknowledgement, it is done with the help of ACK message. If message transfer has not done successfully NON message will be returned. CoAP follows DTLS [13] to provide secure communications.

3.3 MQTT

MQTT designed by IBM and adopted by OASIS [6] for IoT. It is a light weight protocol for supporting constrained devices. It follows publish/subscribe communication model, fig 3 explains it. It has three important parties i.e. publisher, subscriber and broker. Publisher keeps the information on the broker which is collected from sensors or other devices with specified topic name. Broker is an import entity who manages received messages and provides to the subscribers. Subscribers need to subscribe for a required topic to get the services. Different brokers available in the market like mosquito, HiveMQ, ActiveMQ, RabbitMQ, VerneMQ, JoramMQ, IBM MessageSight and Emqttd. MQTT has three QoS levels QoS 0, 1, and 2 [6, 14]. QoS 0 transfers the message but it does not worry about whether message received by receiver or not. QoS 1 ensure message transfer by accepting a confirmation from the receiver. QoS 2 ensures message delivery by avoiding sending duplicate message. MQTT-SN protocol intended for sensor networks by following UDP. But it supports fewer platforms.

Fig 4: MQTT interaction model.

3.4 Data Distribution Service (DDS)

DDS is publish/subscribe model introduced by Object Management Group (OMG) [9]. It allows direct communication between peer to peer instead of broker involvement. DDS offers dynamic discovery, scalability and interoperability. Dynamic discovery allows subscribers to find out who are the publishers present. Fig 4 explains DDS architecture.

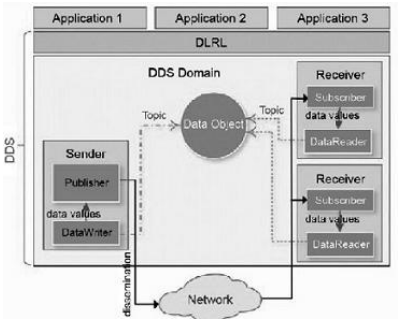

Fig 5: DDS architecture

DDS has main entities Data Reader and a Data Writer [15], topic, publisher, subscriber Domain, Domain Participant. DDS maintains a Global Data Space (GDS) which is accessed by clients store and read data. Publishers uses Data Writer to send data to the GDS. Subscribers uses Data Readers to receive data from GDS. By default DDS follows UDP but it also supports TCP. DDS uses TLS to provide secure communication if TCP is the transport protocol, DTLS for UDP.

3.5 AMQP

AMQP is an interoperable protocol introduced by OASIS [8], which follows publish/subscribe communication model. Because of its interoperability feature, it supports wide range of heterogeneous applications. It has 3 important thins i.e. Exchange, Message queue and Binding.

Exchange: It routes messages to respective queues which are received from publishers.

Message queue: It keeps message in the queue until received by the clients.

Binding: It describes the state between message queue and the change.

TCP protocol is being used by AMQP to provide reliable communication. It has three QoS levels as MQTT. For security it uses TLS for encryption SASL (Simple authentication and security layer) for authentication.

3.6 XMPP (Extensible Messaging and Presence Protocol)

XMPP is an instant message exchange protocol defined by IETF [10]. It is Extensible Mark-up Language and text based protocol. XMPP follows client/server communication model, newer versions support publish/subscribe model. It uses TCP to provide reliable message exchanges. TLS is the inbuilt protocol for security. It has standard syntax i.e. <presence>, <message>, and <iq/> (infor / query). Message has message content and title to exchange data between XMPP devices. Presence describes status updates. Iq gives more information regarding servers.

Protocol	Req.-Rep.	Pub.-Sub.	Standard	Transport	QoS	Security
REST HTTP	✓		IETF [41]	TCP	-	TLS/SSL
MQTT		✓	OASIS [42]	TCP	3 levels	TLS/SSL
CoAP	✓	✓	IETF [43]	UDP	Limited	DTLS
AMQP	✓	✓	OASIS [44]	TCP	3 levels	TLS/SSL
DDS		✓	OMG [45]	TCP/UDP	Extensive	TLS/DTLS/DDS sec.
XMPP	✓	✓	IETF [46]	TCP	-	TLS/SSL
HTTP/2.0	✓	✓	IETF [47]	TCP	-	TLS/SSL

Table 1. Comparison of IoT application layer protocols features

3 INTEROPERABILITY

Interoperability is a technique which makes communication among heterogeneous devices and software applications made from different vendors. It is essential for smart things which are made from different manufacturers. Interoperability has four dimensions technical, syntactical, semantic and organizational interoperability.

Technical interoperability:

It describes interoperability problems related to hardware, software and protocols used for communication.

Syntactical interoperability:

It explains common data format. When data exchanges among heterogeneous devices, data format must be unique.

Semantic interoperability:

It describes common standards among various organisations. For example two temperature sensors, one sensor displays temperature in Celsius, other one displays temperature in Fahrenheit.There is no compatibility between these two sensors. So there is a need of common semantics for these two sensors.

Organisational interoperability:

All industries must maintains identical pattern of organisation.

4 VISUALIZATION OF INTEROPERABILITY PROBLEM BETWEEN COAP AND MQTT USING NODE RED

This section explains an interoperability issue using java script based visualization tool called node red.

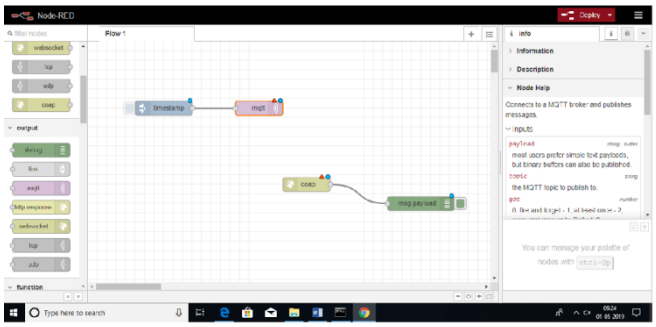

Fig 6: Node-red editor

Fig 6 explains inter communication between CoAP and MQTT protocol. Sender uses MQTT protocol to send information to the server using particular URI. Receiver uses MQTT to receive information from particular topic. MQTT require broker to exchange information between sender and receiver. There is no compatibility between these two protocols that is why message do not be exchanged. It requires a middleware technologies. Few solutions have proposed for semantic interoperability like FIESTA. If we use the same protocols (CoAP or MQTT etc.,) both sender and receiver side message will be transferred without any problem. When we use different protocols transmission is not possible due to in compatibility among application layer protocols. Because different protocols use different communication models and different message transfer syntaxes.

5 CONCLUSION AND FUTURE SCOPE

Interoperability is a technique which makes communication among heterogeneous devices and software applications made from different vendors.Technical interoperability is an important thing at application layer level since various protocols are available. Establishing

communication among heterogeneous protocol is a big challenge. FIESTA is popular interoperability solution to solve sematic interoperability problems. In future there is a need for developing a solution which allows communication among heterogeneous protocols like CoAP, MQTT, DDS, AMQP and XMPP. DDS is an interoperable protocols which uses publish subscribe model.

REFERENCES

1. O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes. 2016. Operating systems for low-end devices in the internet of things: A survey. IEEE Internet of Things Journal 3, 5 (Oct. 2016), 720–734. DOI:http://dx.doi.org/10.1109/JIOT.2015. 2505901W.-K. Chen, *Linear Networks and Systems* (Book style).Belmont, CA: Wadsworth, 1993, pp. 123–135.
2. Y. Xu, V. Mahendran, W. Guo, and S. Radhakrishnan. 2017. Fairness in fog networks: Achieving fair throughput performanceinMQTT-basedIoTs.InProceedingsofthe201714thIEEEAnnualConsumerCommunicationsNetworking Conference (CCNC'17). 191–196. DOI:http://dx.doi.org/10.1109/CCNC.2017.7983104 B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.
3. P. Sethi and Smruti R. Sarangi. 2017. Internet of things: Architectures, protocols, and applications. Journal of Electrical and Computer Engineering (2017), 1–25. DOI:10.1155/2017/9324035 J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.
4. AlessioBotta,WalterdeDonato,ValerioPersico, andAntonioPescapé.2014.Ontheintegrationofcloud computing and internet of things. In Proceedings of the 2014 International Conference on Future Internet of Things and Cloud (FICLOUD'14). IEEE Computer Society, 23–30. DOI:http://dx.doi.org/10.1109/FiCloud.2014.14 Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interfaces(Translation Journals style)," *IEEE Transl. J. Magn.Jpn.*, vol. 2, Aug. 1987, pp. 740–741 [*Dig. 9th Annu. Conf. Magnetics* Japan, 1982, p. 301].
5. G. Banavar, T. Chandra, B. Mukherjee, J. Nagarajarao, R. E. Strom, and D. C. Sturman. 1999. An efficient multicast protocol for content-based publish-subscribe systems. In Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (Cat. No.99CB37003). 262–272. DOI:http://dx.doi.org/10.1109/ICDCS.1999.776528(Basic Book/Monograph Online Sources) J. K. Author. (year, month, day). *Title* (edition) [Type of medium]. Volume(issue). Available: http://www.(URL)
6. AndrewBanksandRahulGupta(Ed.).29October2014. MQTT Version 3.1.1.OASISStandard.http://docs.oasis-open. org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. (Journal Online Sources style) K. Author. (year, month). Title. *Journal* [Type of medium]. Volume(issue), paging if given. Available: http://www.(URL).
7. Z. SHELBY, K. HARTKE, AND C. BORMANN. 2014. THE CONSTRAINEDAPPLICATION PROTOCOL (COAP). RFC 7252. RFC EDITOR. HTTP://WWW.RFC-EDITOR.ORG/RFC/RFC7252.TXT.
8. OASIS. 29 October 2012. Advanced Message Queuing Protocol (AMQP) Version 1.0. OASIS Standard. http://docs. oasis-open.org/amqp/core/v1.0/os/amqp-core-overview-v1.0-os.html.
9. OBJECT MANAGEMENT GROUP (OMG). 2015. DATA DISTRIBUTION SERVICE (DDS) VERSION 1.4. (MARCH 2015), 1–20.
10. P. Saint-Andre. 2004. Extensible Messaging and Presence Protocol (XMPP): Core. RFC 3920. RFC Editor.
11. C. Severance. 2015. Roy T. Fielding: Understanding the REST style. Computer 48, 6 (June 2015), 7–9. DOI:http://dx. doi.org/10.1109/MC.2015.170
12. W.Shang, Y.Yu,R.E.Droms,andL.Zhang.2016. Challenges in IoT Networking via TCP/IP Architecture.NDNProject, Technical Report. NDN-00382 (2016), 7.
13. E. Rescorla and N. Modadugu. 2012. Datagram Transport Layer Security Version 1.2. RFC 6347. RFC Editor. http://www.rfc-editor.org/rfc/rfc6347.txt.
14. J. E. Luzuriaga, J. C. Cano, C. Calafate, P. Manzoni, M. Perez, and P. Boronat. 2015. Handling mobility in IoT applications using the MQTT protocol. In Proceedings of the 2015 Internet Technologies and Applications (ITA'15). 245–250. DOI:http://dx.doi.org/10.1109/ITechA.2015.7317403
15. G. Farabaugh, G. Pardo-Castellote, and R. Warren. 2005. An introduction to DDS and data-centric communications. Real-Time Innovations.August (2005

AUTHORS PROFILE

V. Tirupathi Assistant professor Computer Science and Engineering S R Engineering College, Warangal. He has 13 years of teaching experience. His research interests are Internet of Things (IoT).

Dr. K Sagar professor Computer science and Engineering CBIT Hyderabad. He has 21 years of teaching experience. His research interests are Soft computing, Machine learning, Advanced Operating systems.