I.J. Wireless and Microwave Technologies, **2015, 6, 1-10**
Published Online November 2015 in MECS(http://www.mecs-press.net)
DOI: 10.5815/ijwmt.2015.06.01

Available online at ***http://www.mecs-press.net/ijwmt***

CoAP Protocol for Constrained Networks

Manveer Joshi [a], Bikram Pal Kaur [a]

[a] *Deptt. of IT, Chandigarh Engineering College, Landran, Mohali, Punjab, India*

Abstract

This paper discusses the Constrained Application Protocol (CoAP), an application layer protocol introduced by Internet Engineering Task Force (IETF) Constrained RESTful environment (CoRE) Working Group for use with low power and resource constrained nodes in the internet of things. The paper discusses the CoAP architecture supported with suitable examples. CoAP protocol message format, types, methods and security feature is also discussed. In this paper, we present a CoAP based car parking system implemented using Contiki, an open source operating system for resource constrained device. The Cooper (Cu) CoAP user agent for Firefox browser is used for handling CoAP URI scheme.

Index Terms: CoAP, TCP, HTTP, UDP.

1. Introduction

In tomorrow's world each object will be identified by its IP, thanks to the enormous address space provided by IPv6, thus allowing interconnection and addressing of each object. The internet will become Internet of Things (IoT) in this case providing interaction between different objects, devices and users. Atzori et al., 2010 defined IoT as: Internet of Things involves communication among many constrained devices such as wireless sensors, smart sensors over the Internet. It features among most promising research areas in wireless area today. Internet of Things, abbreviated as IoT, is a network of smart devices (sensors, actuators) accessible over the Internet

To communicate among devices over internet web services constitutes an essential part and can be realized primarily using RESTful or Remote Procedure Call (RPC) approach. In the first cases resources such as pictures, video files, Web pages, business information, or anything are managed by HTTP protocol whereas in the latter case it is managed by SOAP protocol. The RESTful approach offers advantages of simple, less parsing complexity, low overhead, statelessness, and offers tighter integration with HTTP. However, RESTful web services have their own share of problems in implementation over constrained wireless devices due to the

* Corresponding author
E-mail address:

protocols and payload formats used to realize them. The problems of large overhead of HTTP headers, TCP performance degradation over lossy links with no multicast support, high parsing complexity of XML if used as payload and inappropriate pull model for sleeping sensors.

Therefore, over the past few years to support constrained devices, Constrained RESTful Environment working group is developing CoAP. Shelby et al., 2013, 2014 defined CoAP as specialized web transfer protocol offering offer simplicity, low overhead and machine-to-machine communications (M2M). CoAP being an application layer protocol interfaces with lower layers of OSI or TCP/IP protocol suite to give reliable connection over network. The CoAP layering model suggests UDP as transport layer but to provide reliability it uses subset of full featured transport layer. CoAP uses Datagram Transport Layer Security for providing communication security but it is an optional feature. In the context of machine to machine model as applicable to constrained networks, CoAP offers comparative features to HTTP while giving advantage of simplicity and low design overheads as suggested in Colitti et al., 2011 and Lerche et al., 2012. The result is low bandwidth, most important need of machine to machine communication. The CoAP protocol uses interactive model between application endpoints which is nothing but a request and response model. It supports resource discovery, multicasting, asynchronous message exchanges with a low overhead, parsing complexity, and URL based content-type support, simple proxy, caching capability for constrained devices and machine to machine Applications. In comparison to CoAP the Hyper Text Transfer Protocol (HTTP) is a complicated protocol having features making it unsuitable for resource constrained devices. Also it uses the TCP, a high bandwidth protocol, thus negatively impacting the WSNs available bandwidth as suggested by Shelby, 2010.

2. Basic Framework of CoAP

This section discusses CoAP technologies, features, core functions, security issues and basic procedures with the help of several examples.

2.1. Architecture

The CoAP interaction model is similar to HTTP client/server model but the CoAP implementation acts as both client and server in typical machine to machine interactions. Similar to HTTP a CoAP request is sent by a client using a Method Code to request an action on a URI identifiable resource.

The server replies with a Response Code which may include a resource representation. Thus, as suggested by Shelby et al., 2013 CoAP model is essentially client/server architecture enabling the client to request for service from server as needed and the server responds. CoAP request is similar in nature to HTTP but as shown in the figure 1 CoAP interchanges are asynchronous since it uses UDP. The message layer interfaces with UDP layer which formats the data received into a datagram and sends it to the lower layers of the OSI or the TCP/IP Model.

Similarly in the opposite direction, the datagram received by UDP presents it to the application layer in a legible format. Logically CoAP comprises of two-layers: a message layer responsible for UDP communication and reliability (optional), while the other layer is responsible for request or response interactions. CoAP also uses asynchronous message exchange between end points. CoAP defines four types of messages:

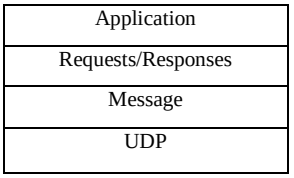

Fig.1. CoAP Layering

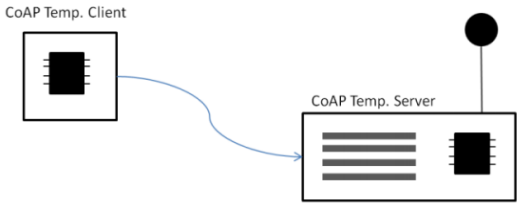

Fig.2. CoAP Client/Server

CoAP defines four types of messages as Confirmable, Non-confirmable, Acknowledgement and Reset. The embedded Method Codes and Response Codes in some of these messages mark them as requests or responses. Below is the example explaining the CoAP Client/Server model, Figure 2, where a temperature sensor is installed in the conference room of an office. The temperature sensor works like a "server" (Thing) which any CoAP based client (another Thing) can query to get the temperature.

2.1.1. Message Format

A binary encoded CoAP message contains a CoAP Header, options and payload. The header decides the available in a Type Length Value (TLV) format (Figure 3). The message fields are defined as follows with all lengths in unsigned integer formats:

Version (Ver): The field length is two-bit showing the CoAP version number, which is one at present.

Type (T): This field is two-bit long showing the message type. There are four message types confirmable, Non-Confirmable, Acknowledgment and Reset represented by bit patterns of 00, 01, 10 and 11 respectively.

Option Count (OC): It is four-bit long field, thus providing maximum of sixteen options after the header.

Code: It is eight-bit long field which shows whether the message is empty (0), request (1-31) and response (64-191). The remaining (192-255) is reserved for future use.

Message ID: This sixteen-bit field is used for detection of message duplication, messages of type acknowledgment/reset and Confirmable.

Payload: The payload carries sensor data or resource representation.

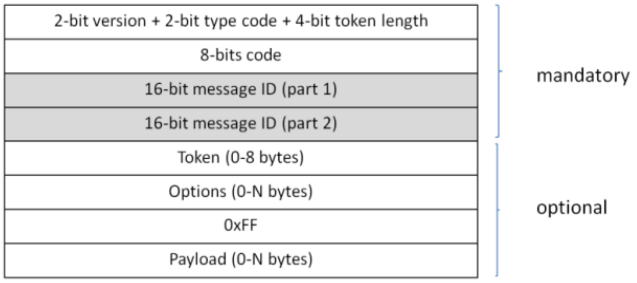

Fig.3. CoAP Message Format

Options: It defines payload message type with available options are Proxy-Uri, Uri-Host, Uri-Port, Location Path, Max-Age, Uri Path Max-Age, Uri Path, Uri Query, and Token Accept.

2.1.2. Message Type

As the CoAP messages are similar to request or response model used in HTTP. The code field of the CoAP

header defines four message types. The requests are carried in Confirmable and Non-confirmable messages whereas responses in these as well as piggybacked in Acknowledgement messages. These message types are as follows:

Confirmable (CON): This message type requires response that is once message is sent; the receiver must confirm the message receipt.

Non- Confirmable (NON): This message does not require response that is once message is sent, the receiver does not need to confirm the message receipt. Thus implying unreliable message type.

Acknowledgment (ACK): This message type is received in response to CON message confirming the latter reception

Reset (RST): This message is sent in case of an error in message, message is not understandable or receiver is not interested in communication with sender.

2.1.3. CoAP Methods

CoAP makes use of GET, PUT, POST, and DELETE methods in a same manner to HTTP and are used to manipulate the resources. Since the basic set of request methods is similar in both HTTP and CoAP, a CoAP request on HTTP resource is similar to one on a CoAP resource. A "405 Method Not Allowed" Response Code should be generate in response to a unicast request with an unknown or unsupported Method. As CoAP methods are similar to HTTP, they exhibit the similar properties of safe (only retrieval) and idempotent (same effect at each invoke) as HTTP. The GET method is safe while GET, PUT and DELETE methods mist be performed in idempotent manner. The POST method is not idempotent because the URI embedded in the request indicates the resource that will handle the enclosed body, which can be used for data processing, a gateway to other protocols and it may create a new resource as a result of the POST. Different CoAP methods are:-

GET: The GET method is used to retrieve resource information identified by the request URI. In response to GET method success a 200 (OK) response is sent.

POST: The POST method creates a new subordinate resource under the parent URI requested by it to server. On successful resource creation on the server, a 201 (Created) response is sent while on failure a 200 (OK) response code is sent.

PUT: The PUT method updates or creates the resource identified by the request URI with the enclosed message body. The message body is considered modified version of a resource if it already exists at the specified URI otherwise a new resource with that URI is created. A 200 (OK) response is received in former case whereas a 201 (Created) response is received in later case. If the resource is neither created nor modified then an error response code is sent.

DELETE: The DELETE method deletes the resource identified by the requested URI and a 200 (OK) response code is sent on successful operation.

2.1.4. Server Discovery

Client discovers server with the knowledge of server URL consisting of server machine address with port number at which CoAP server resides. The default port is 5683 as defined in Shelby, 2013. Therefore, the client needs to: set up a UDP connection with the server, send a GET request to the server over the given URL path (coap://www.example.com:5683/ sensor/temperature) and get a response.

2.1.5. Resource Discovery

CoAP uses standard methods for resource discovery. Similar to standard methods, servers have a list of available resources (with metadata about them) in the application/link-format to allow a client for discovering them and their media type. Predefined macro defines resources. Each resource needs name, path, interface

description, resource type and the code. For periodic resources the actuator must know the period. A callback function performs the needed action.

2.1.6. Observe

Resource observation in CoAP is simply an extension of HTTP request. The CoAP GET resource checks for set condition of observe flag. The flag sets on receiving a CoAP GET resource. Client does not need to request repeatedly while the server responds by seeing the change in parameters. This enables servers to communicate the state changes as needed by the client. Both server and client have authorization to end the observation.

2.1.7. Security

DTLS and IPSec are two methods defined in CoAP specification for security by providing encryption, authentication, integrity and replay protection features. These features are minimum that should be supported by networks in which CoAP operates. However CoAP uses UDP at transport layer and DTLS for security. The DTLS protocol is an improved version of the widely used Transport Layer Security (TLS) protocol. The major difference is that DTLS runs on top of UDP instead of TCP to secure UDP applications. DTLS provides authentication, data integrity, confidentiality and automatic key management. It also supports a wide range of different cryptog raphic algorithms, which makes it a potential security protocol. Alghamdi et al., 2013 and Brachmann et al. 2011discussed the security services provided by the DTLS and IPSec Protocol and is shown in Table 1.

Table 1. IPSec vs. DTSL

Service	IPSec	DTLS
Access Control	No	No
Authentication	Yes	Partially- Server Only
Non Repudiation	Depends on authentication method	Depends on authentication method
Confidentiality	Yes	Yes
Commu. Security	Yes	Yes
Integrity	Yes	Yes
Privacy	No	No
Availability	Not Fully	Yes

3. Applications of CoAP

3.1. Basic CoAP Based System Setup

The lowest level of CoAP based system consists of machines that is sensors and actuators that measures or take actions. These sensor or actuators forms a small network to interact with the outer world over CoAP. There may be need to setup a proxy which finally sends data to an HTTP server. The Figure 4 below shows the setup as shown by Sharma. The circles marked with “C” show CoAP based machines.

3.2. Real Time Condition Based Monitoring in Smart Grid

A Smart Grid is an intelligent power generation, distribution and monitoring system. It uses the modern information and communication technologies to gather and act on data. Today, multiple sensors enclosed in a unit can be placed on the transformer. The unit, then in turn, sends the data to the data centres by different

means (PLC, GPRS, and Ethernet). They send the data over 6LowPAN with CoAP to the edge router and proxy combination. This will standardize the way manufacturers create the sensors.

3.3. Building Automation

Inside large building complexes there may be many sensors and actuators that are principal candidate for replacement with CoAP based devices. The temperature control, garbage control and automatic light control are some examples among many examples. Smart City envisages automation of many activities which also involves buildings. To take an automated garbage collection example, garbage collection takes place from all floors of the building and discarded to a central facility automatically, thus avoiding human intervention. Smart home can also use this technology because of the fact that many sensors in home are low bandwidth sensors (temperature sensor, alarm, light sensor) as well as large bandwidth sensors (cameras, security systems) and user wishes to control them distantly with the help of device browser only. Many companies have launched products based on similar lines. Bergmann et al., 2012 proposed a CoAP gateway for smart home. They used CoAP and proprietary FS20 protocol for home automation and showed the feasibility of CoAP for such scenario.

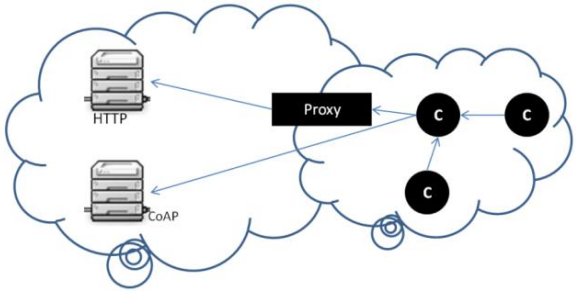

Fig.4. Basic CoAP System Setup

3.4. Defense Equipment

Battle tank today has thousands of sensors each connected with wires. It is good to replace these tiny sensors with another tiny set of sensors based on CoAP. Intruder detection system consists of many tiny sensors to detect any intrusion. These sensors hop their data from one another to server, which connects to a secure network running standard Internet protocols. Some of these sensors bandwidth is few bytes or one byte sometimes, therefore it becomes essential that protocol selected should have low overhead. Thus making CoAP as preferred choice, moreover it also gives interface to http. However, CoAP uses duty cycling mechanism for power saving, an essential requirement for small nodes to have a long life.

3.5. Aircraft Equipment

Today, a commercial aircraft contains miles and miles of wiring just connecting many sensors and actuators with each other. Smart CoAP based sensors and actuators may help to save weight in the aircraft, making it more efficient.

3.6. Factory Instrumentation

Many manufacturing factories centrally manage and control various kinds of instruments for measuring different parameters. Sensors used in these applications are low bandwidth sensors on which standard protocols

do not work because of complexity and cost. In such case a low bandwidth protocol like CoAP is effective in managing cost and providing simplicity.

4. Related Work

Andrea et al, 2014, proposed urban Internet of things concept applicable to smart cities. He presented a survey of the enabling technologies, protocols, and architecture to be used for an urban IoT. He also discussed the Furthermore technical solutions and guidelines adopted in the Padova Smart City project. Bergmann et al., 2012, discussed possibilities of using CoAP for smart homes. He showed the basic design concept by using CoAP and the FS20 protocol for home automation. Krimmling et al., 2014, discussed security issues arising due to usage of CoAP in smart cities. He evaluated intrusion detection techniques designed for smart public transport application that is using CoAP. Rajesh et al., (2015 controlled multiple appliances using CoAP and wireless embedded home gateway. The gateway used CoAP for data transfer between different home appliances and the client through internet.

5. CoAP based Car Parking System: An Example

A smart transport system consists of real time monitoring and control of traffic for efficient use of existing infrastructure. Better management of parking facilities will ease congestion in urbanized areas as cars get off the street easily and parked in the reserved slots. Web based parking is one of the effective method to achieve the goal. Car parking is a tedious task for lack of space in congested cities of present world. The example application can act as a method to find the available parking in the needed area. The user can reserve parking slot while sitting at distant place from parking space. The need is an Internet connection and a browser with CoAP user agent add-on. Such system can help make our lives simpler and our cities better.

There exist different models for parking monitoring. The parking model chosen for this example is primitive one. The method employed is to reserve parking lots on first come first fill basis. This means parking lots is assigned serially. Figure 5 depicts a simple wireless sensor network model. It consists of parking lots, car, buzzer, RFID tag and devices with Internet connectivity. The user establishes a connection with the car parking system using browser over Internet. After establishment of connection user enquires for available free parking spaces. Based on the query response user can decide his action. This way it helps user in decision making. The data transmission from internet to that of sensor uses Constrained Application Protocol. After assignment of parking slot user can park his car according to schedule. The buzzer sound ensures correct parking by the user at assigned slot.

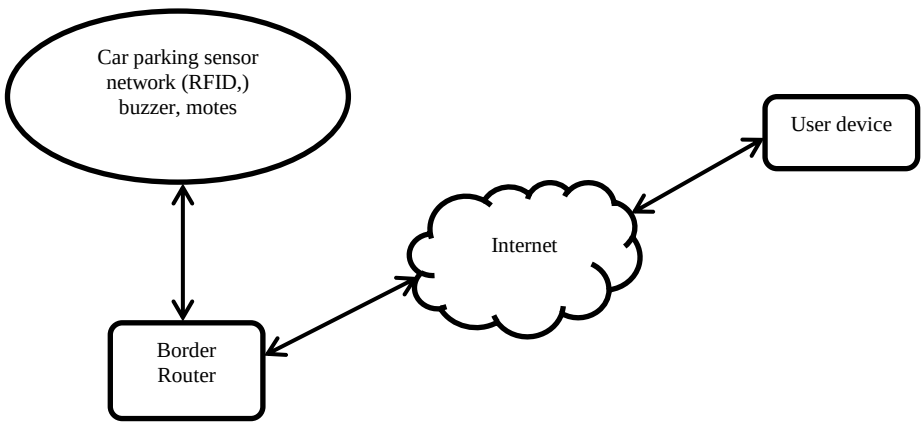

Fig.5. Car Parking Wireless Sensor Network

A CoAP application is similar to restful applications that use Uniform Resource Identifiers (URI) for addressing. A CoAP URI used in this example is coap://cooja2:5683/. The user enters URI in browser and requests for a resource. The browser displays the result. There exists different CoAP solution for constrained devices such as ISense, Wiselib, Arduino and Contiki. Among above Contiki is selected solution for application. Contiki is an open source operating system for resource-constrained devices. It supports a wide range of open source low-power wireless devices such as Arduino, MSP430 and sky motes. Contiki uses Cooja, a Java-based simulator designed for sensor networks running the Contiki sensor network operating system. Though Cooja simulator source code is in Java but it allows sensor node software to write in C language. The wireless nodes used in application is sky node. The CoAP client used in browser is Cooper. There are three types of nodes used in example application: server, border router and client. In the example network there are many clients supported by border router and server. The server node is restful server which uses REST layer for server side application development. Border routers exist on network edge and connect different networks. They data between a WSN network and an external IP network routes with their help. Border Router keeps radio turned on, enabling of it helps in connection between that of client as well as server to that of CoAP web Address. Border Router has the same stack and fits into mote memory. A CoAP client cycles through four resources on event detection such as GET, PUT, PUSH, and DELETE. It connects to the server with multi-hop topology. The Copper add on receives data from motes and displays it in a suitable format on user device web browser.

The resources shown in example are: EnterYourDetails, KnowYourVehicleParkingSlot, Parking Information. Enter YourDetails asks about vehicle number and entry or exit. For example if a user wants to park his car, the format is “invehicleid” and for exit “outvehicleid” Figure 6. KnowYourVehicleParkingSlot provides information about vehicle parking slot number. Parking Information gives number of free slots to user. The flowchart shown in Figure 7 depicts parking slot reservation and release. As shown in flowchart, a buzzer plays sound on wrong parking. This is done with the help of RFID tag allotted to car, matching of which with corresponding sensor plays the buzzer. The parking staff or user can take corrective action on hearing buzzer.

6. Conclusion

This paper presented the CoAP framework. The purpose of CoAP, advantages, needs and major features is discussed. Examples of application programs gave an insight into the CoAP based software and current development. Graphical User Interface of application example discussed with snapshot and flowchart. Today software developers use CoAP extensively in various areas of software development. Many companies are launching tools, frameworks, products, hardware and software in this area. This opens many research areas like security, power optimization, architecture and networking to researchers. Compared to existing HTTP protocol, CoAP gives better quality of service, value-added services and Machine to Machine communication support. Thanks to the extensibility of CoAP over HTTP for solving the constrained devices issues.

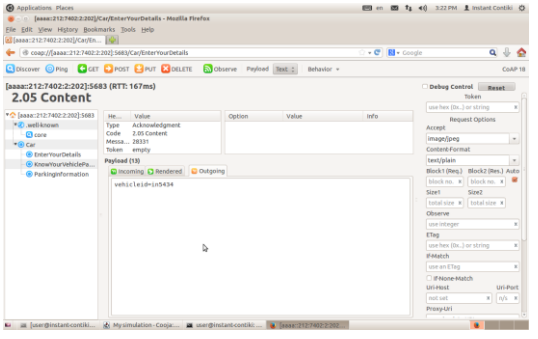

Fig.6. Screenshot of Enter Your Details Resource

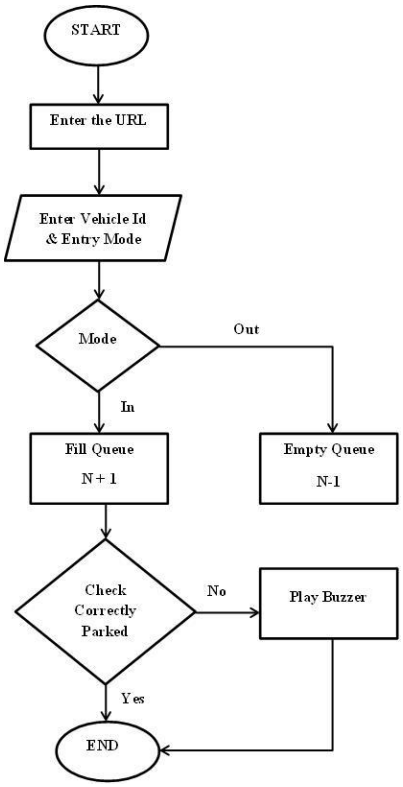

Fig.7. Flowchart for Vehicle in or Out Process

References

[1] Atzori L, Iera A, Morabito G. The Internet of Things: A Survey. Computer Networks: The International Journal of Computer and Telecommunications Networking 2010 Oct; 54(15): 2787–2805.

[2] Constrained RESTful Environments(CoRE)Working Group. Available at: https://datatracker.ietf.org/wg/ core/charter/.

[3] Shelby Z, Hartke K. Constrained Application Protocol (CoAP). Draft-ietf-core-coap-18. [2013-06--28] http://tools.ietf.org/html/draft-ietf-core-coap-18.

[4] Shelby Z, Hartke K, Bormann C. The Constrained Application Protocol (CoAP). draft-ietf-core-coap 2014 June.

[5] Colitti W, Steenhaut K, De Caro N, Buta B, Dobrota V. Evaluation of Constrained Application Protocol for Wireless Sensor Network. Proceedings of 18th IEEE International Workshop of Local and Metropolitan Area Networks (LanMan), Chapel Hill, NC, USA, 2011 Oct; 1 - 6.

[6] Lerche C, Hartke K, Kovatsch M. Industry Adoption of the Internet of Things: A Constrained Application Protocol Survey. Proceedings of the 7th International Workshop on Service Oriented Architectures in Converging Networked Environments (SOCNE 2012). Krakow, Poland, 2012 Sept.

[7] Shelby Z. Embedded web services. IEEE Wirel. Commun. 2010; 17: 52–57.

[8] Sharma V. Understanding Constrained Application Protocol. EXILANT Technologies Private Limited.

[9] Alghamdi T A, Lasebae A, Aiash M. Security analysis of the constrained application protocol in the Internet of Things. 2nd Int.Conference on Future Generation Communication Technology (FGCT). 2013.

[10] Brachmann M, Garcia-Morchon O, Kirsche M. Security for Practical CoAP Applications: Issues and Solution Approaches. GI/ITG KuVS Fachgesprch Sensornetze (FGSN), 2011.
[11] Bergmann O, Hillmann K T, Gerdes, S. A CoAP-gateway for smart homes. International Conference on Computing, Networking and Communications (ICNC); 2012: 446 – 450.
[12] Zanella A, Bui N, Castellani A, Vangelista L, Zorzi M. Internet of Things for Smart Cities. IEEE Internet of Things Journal 2014 Feb; 1(1): 22-32.
[13] Bergmann O, Hillmann K.T.,Gerdes, S.. A CoAP-gateway for smart homes. International Conference on Computing, Networking and Communications (ICNC) 2012; 446 – 450.
[14] Krimmling J, Peter S. Integration and evaluation of intrusion detection for CoAP in smart city applications. IEEE Conference on Communications and Network Security (CNS), 2014 Oct; 73-78.
[15] Rajesh K R, Bindyashree CA. Multiple Appliances Controlling and Monitoring System based on wireless Embedded Home Gateway. Int. Journal of Innovative Research in Computer and Comm. Engg. 2015 April; 3(4): 2872-2877.

Author(s) Profiles

Manveer Joshi is M.Tech (IT) student of Chandigarh Engineering College, Landran, Mohali. She has more than 5 years of industrial experience.

Dr.Bikram Pal Kaur is a Professor Chandigarh Engineering College, Landran,Mohali. She holds the degrees of B.Tech.,M.Tech,M.Phil.,Ph.D in the Computer Engg from Punjabi University, Patiala. She has more than 21 years of teaching experience and served many academic institutions. She is an active Researcher who has supervised many B.Tech, M.Tech/ Research Projects, M.Tech. Disserations and also guiding Ph.D to seven scholars . She has contributed more than 38 articles in various national/ international conferences and 34 papers in research Journals. Her areas of interest are Information System, ERP and Parallel Computing.

How to cite this paper: Manveer Joshi, Bikram Pal Kaur,"CoAP Protocol for Constrained Networks", IJWMT, vol.5, no.6, pp.1-10, 2015.DOI: 10.5815/ijwmt.2015.06.01