International Journal of Academic Research and Development
ISSN: 2455-4197
Impact Factor: RJIF 5.22
www.academicsjournal.com
Volume 3; Issue 3; May 2018; Page No. 249-253

Network security using constrained application protocol (CoAP)

Amol Kadam[1*], Srija Srivastava[2], Vijay Suryawanshi[3], Nikhil Thorat[4], Abhinav Parashar[5]
[1]Assistant Professor, Department of Computer Engineering, Bharati Vidyapeeth Deemed University College of Engineering, Pune, Maharashtra, India
[2-5] Department of Computer Engineering, Bharati Vidyapeeth Deemed University College of Engineering, Pune, Maharashtra, India

Abstract
The Internet of Things (IoT) has emerged as the key networking paradigm for supporting the connectivity of massively distributed objects and numerous simultaneous applications. The Constrained Application Protocol (CoAP) is designed to meet the requirements for IoT data transmission among constrained nodes. The lean design of CoAP enables it to additionally meet the needs of the data transmission in dynamic network environments. In this paper, we conduct an emulation-based quantitative performance assessment of CoAP in comparison with HTTP, assessing data transmission based on key characteristics of dynamic network environments and the designed scenarios. We also designed scenarios and evaluate the performance of investigated protocols using real-world IoT datasets. Our experimental results demonstrate that CoAP performs better than HTTP for data transmission in the dynamic network environments with respect to delivery rate, delay, and overhead. In addition, we analyze the impact of features in dynamic network environments on the performance of data transmission protocols with respect to success rate, delay and overhead, as well as discuss some further extensions for future research.

Keywords: CoAP, dynamic networks, performance evaluation, emulation, IoT, applications and service

1. Introduction
The Internet of Things (IoT), the immensely distributed connection of sensing, actuating, and computing objects, has given rise to the need for networking technologies that support the simultaneous connectivity of these devices, enabling smart applications, including smart grid, situational awareness in the battle field, smart transportation, smart healthcare, smart cities, public safety, etc. A key idea of IoT is that microchips and sensors with network connectivity are embedded into various objects, enabling dispersed data collection and exchange with the support of modern information communication techniques [1]. As a result, these physical objects are able to be observed and controlled remotely through the network. Despite the rapid and significant reduction in scale, the single IoT device itself is typically limited by electrical power capacity, memory, and computing ability. As a result, traditional protocols, such as HTTP, may not be able to perform efficiently on a network that consists primarily of these constrained devices, as these limitations increase the difficulty of data transmission.

To address this issue, the Constrained Application Protocol (CoAP) [10] has been proposed. Generally speaking, CoAP is designed to meet the needs of constrained data transmission by reducing overhead, as well as through other features, including resource discovery, multicast, asynchronous message exchange, etc. There have been a number of research efforts devoted to the application and evaluation of CoAP in the context of IoT applications, including crowd-sensing infrastructures, smart cities, vehicle tracking systems, and many others. In these investigations, CoAP was also evaluated for its scalability, resource consumption, security, proxy, group communication, and congestion control.

In this study, we carry out the quantitative evaluation of CoAP transmission over UDP in dynamic network environments (in the form of MANET as an example) through emulation. HTTP transmission over TCP has also been emulated as the comparative baseline scheme. The target transmission models have been determined via our analysis on the features of each protocol with respect to its specification. Based on the analysis, CoAP Confirmative, CoAP Non-confirmative, and HTTP were selected for the evaluation in 1-to-1 and 1-toN communication. An additional protocol, CoAP multicast, was also evaluated in 1-to-N communication. We conduct theoretical analysis of the impact of features in dynamic network environments on the performance of data transmission protocols with respect to success rate, delay and overhead. The Common Open Research Emulator (CORE) [12], a network emulation tool, was leveraged to carry out the evaluation. The emulation scenarios were designed to evaluate the following key characteristics of the dynamic network environment: distance, mobility, and disruption. The scenarios tested each protocol under the singular influence of each characteristic, as well as a more realistic scenario that combined all three characteristics, in order to observe and compare overall performance. The experimental results show that CoAP has a significant advantage in speed, overhead, and reliability in transmitting small volumes of data in the dynamic network environments when using Confirmative messages.

2. Background

This section reviews the key points of the protocols, concepts, and tools that are involved in the protocol evaluation in this paper.

2.1 CoAP

The Constrained Application Protocol (CoAP) is designed for use with constrained nodes and networks. Consistent with HTTP, CoAP follows the request/response model and the CoAP resources are identified by the URI. CoAP has four methods: GET, POST, PUT, and DELETE, which enable the basic CRUD (primitives) operations on the resource by the clients. CoAP also defines request and response messages. Distinct from HTTP, the CoAP message is bitwise structured. The message header includes bits of Version, Type, Token Length, Message ID, Token, Option, Payload Marker, and finally, the Payload. Since Token and Option are optional, the minimum size of the header of a CoAP message is only 5 bytes. A 2-bit field "Type" is defined in the message header, indicating four types of messages: Confirmative (CON), Nonconfirmative (NON), Acknowledgement (ACK), and Reset (RST). This feature enables CoAP to choose the transmission reliability from the application layer. Unlike HTTP over TCP, CoAP messaging is bound to UDP, which does not provide a retransmission mechanism. The CoAP request can be sent on the default CON mode so that an exponential retransmission mechanism would ensure the reliability of data transmission. In addition, the request can be set to the NON mode with no reliability ensured. Other features are defined in CoAP, such as resource discovery, observe and notification [14], blockwise transferring [15], group communication [16], and Proxy (CoAP-to-CoAP, CoAP-to-HTTP, and HTTP-to-CoAP). To implement CoAP, we have leveraged the open source project "Californium" [17], which is a java based project of the Eclipse Foundation that provides the framework with protocol implementation. Using the source code and class libraries, we can implement lightweight CoAP servers and clients to deploy in our emulated nodes, and perform the CoAP request/response procedures.

2.2 CORE

The Common Open Research Emulator (CORE) was developed by Boeing and supported by the Naval Research Laboratory [12]. CORE is a python framework that provides a GUI for users to build and emulate network topology. The emulated environment consists of a number of lightweight virtual nodes and methods for connecting the nodes. A number of protocols, applications, and scripts (e.g., mobility patterns) can run on the emulated nodes, which can in turn form an emulated network. This emulated network is additionally able to be connected to real networks using the provided interface. In our investigation, we use CORE to establish the dynamic environment (in form of MANET as an example) for assessing the performance of CoAP.

3. Our Approach

To evaluate the selected protocols, it is critical to analyze the features of protocols and the operating environments to determine the scope of the evaluation, the metrics, and the key indicators. In the following, we introduce the design rationale, the CoAP protocol features, the features of dynamic networks, and scope of experiments in detail.

3.1 CoAP

CoAP is application layer protocol especially designed for constrained devices. CoAP uses UDP to run as underlying protocol.

CoAP provides REST interface to provide efficient communication among the devices. To protect CoAP communication, DTLS has been selected as basic security protocol. CoAP using DTLS security is termed as secured CoAP (CoAPs) like the TLS secured HTTP as HTTPs. CoAP uses Universal Resource Identifier (URI) to access the resources on destination device. CoAP protocol uses "coap" URI scheme. CoAP securely accesses the web resource on destination device as follows:

CoAP is simply a request-response type protocol and provides both types of communication viz. reliable and unreliable. Devices using the CoAP may act as client, server or both. The reasons that a new protocol is defined for constrained IP networks, instead of simply reusing HTTP, is to greatly reduce overhead in implementation complexity and to reduce the bandwidth requirements. Such data reduction also helps to increase reliability by reducing link layer fragmentation and reduce latency in typical low-power lossy wireless networks, such as IEEE 802.15.4. assessment, we focus on a dynamic network environment, which can be characterized by factors such as distance, mobility, and disruption, Via a combination of theoretical analysis and extensive experiments in CORE, we validate the impacts of factors in the dynamic network environment on the performance of data transmission protocols with respect to key performance metrics: success rate, delay, and overhead. In the design of scenarios, to validate the effectiveness of the CoAP protocol in comparison with HTTP, we first consider the basic topologies with controlled factors (Group 1 and Group 2) to investigate the impacts of the individual factors on the performance of protocols. Then, we focus on typical IoT topologies that are combined with a variety of factors using simple simulated data (Group 3 and Group 4). After that, we consider real-world datasets using typical IoT topologies (Group 5 and Group 6). We also summarize the scenarios in which CoAP is recommended to better support applications, as well as future research avenues.

3.2 DTLS

DTLS is complete security protocol that performs key exchange, authentication and securing application data by using algorithms and negotiated keying material. DTLS contains the two layers, lower layer and upper layer2. Lower layer contains the record protocol and upper layer contains one of the three protocols, Handshake, Alert and Change Cipher Spec or data. The handshake process uses Change Cipher Spec to indicate record protocol should protect subsequent messages by using cipher suite. Alert protocol is used to communicate error message between nodes. Record header contains the fragment field and content type. Fragment field contains one of three handshake protocol, alert protocol,

Change Cipher Spec protocol or data based on value contained in content type. The record header protocol has the responsibility of protecting upper layer protocols. DTLS handshake protocol is chatty and contains number of message exchanges in asynchronous manner. The handshake messages are organized in flights and used to exchange the information like cipher suites, security keys and compression methods.

3.3 Drawback of Existing Systems

- Existing system is not secure
- We cannot prevent our private data with existing system
- Anyone can easily access our private network with simple hacking tricks like sql injection or man-in- middle attack
- We cannot restrict authorized users to denied website which we don't want to give access

4. Proposed System

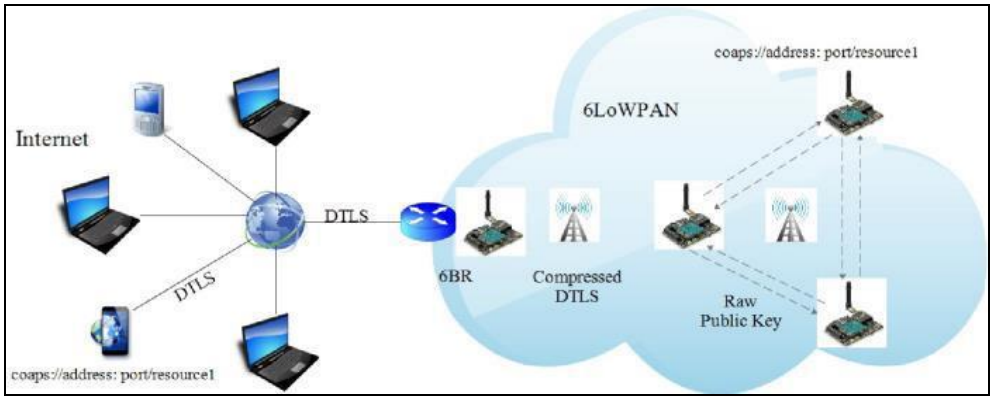

Fig 1: Propose System

Fig. shows typical network setup for IoT, in which 6LoWPAN network containing CoAPs enabled devices are connected through 6LoWPAN Border Router (6LBR) with conventional Internet. As shown in figure header compression is applied in the 6LoWPAN network only, between constrained devices and 6LBR. To allow the devices in multivendor environment to authenticate each other, the concept of raw public key has been introduced instead of X.509 certificates10. Further for vehicle motion. The paper will shed light on the features & design of the system.

4.1 Features

The CoAP protocol has the following features -

1. It provides M2M communication in constrained environment.
2. It provides security of data by datagram transportation layer security (DTLS).
3. Asynchronous message exchange.
4. Low header overhead and parsing complexity
5. URI and content type support
6. UDP binding with optional reliability supporting unicast andmulticast requests.

The CoAP is different from other protocols. When compared with HTTP, CoAP is implemented for IoT and M2M environment to send messages over UDP protocol. To compensate for the unreliability of UDP protocol, CoAP defines a retransmission mechanism and provides resource discovery mechanism with resource description.

The MQTT and CoAP, both are designed to consider the long term vision that they need to be used in lightweight environments. Both works well with low power and network constrained devices. So the choice really depends on the application. If a M2M network has to be created where messages must publish from one node to multiple interested nodes, the best choice will be to use MQTT protocol. But if a M2M network has to be created where commands must be sent to Figure 6: Unicast topology in cooja among nodes, CoAP will be the best choice for that. For example to control an AC from the smart phone, CoAP would be the smarter choice.

CoAP Should Be On Priority for the Following Three Factors

- Quality of service with confirmable message
- When multicast support is needed
- Very low overhead and simplicity.

5. Implementation

For implement or test LESS algorithm to secure unicast communication we used contikios with cooja simulator as a Implementation tool. Contiki is open source operation system. It is lightweight OS and implement in C programming language. It is purposely developed and created for the low-power devices in constrained environment . It is connects constraint node to Internet. It provide powerful low power Internet communication. It supports IPv6 and IPv4 standard along with 6LoWPAN, CoAP and RPL. It can be used in commercial, non-business and full source code is simple accessible.

Contiki application are written is C programming language and used cooja simulator for simulation purpose so that network can be emulated before burned into hardware. Cooja is network simulator for Contiki which allowed larger and small network for simulation. Cooja control and analyze contiki system via few function. In front-end interface, cooja used combination of java codes . It is a cross-level simulator, built in Contiki OS. It Provide the simulation on network

level, OS level and machine code level. It is network simulator for contiki which allowed big and tiny network for simulation.

First we create unicast communication in cooja simulator show Figure 2 to topology of unicast communication. In this topology one is client and another one is server and green circle is radio environment for strong communication.

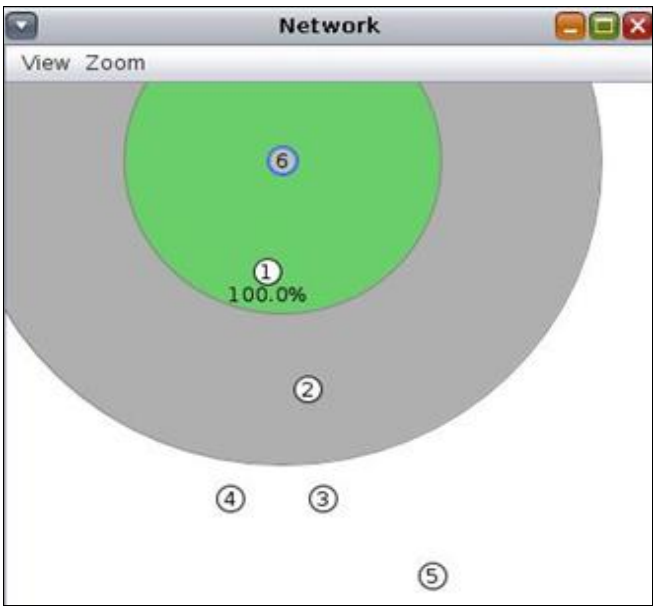

Fig 2: Unicast topology in cooja

Now apply LESS algorithm to client and server both side and below Table 1 show implementation result. In cooja simulator take 850 ms time for session key generation.

6. Conclusion and Future Scope

There are three major components for implementing IoT on different applications: Security, Privacy and Trust. While increasing the growth of IoT, security is more important for reliable data transferred among the billions of smart objects. In these research, we concentrate on CoAP protocol application layer protocol on IoT devices. Having light weight and consume low energy, CoAP is used on many applications of IoT. To secure data transferred, CoAP combined with DTLS protocol named as Datagram Transport Layer Security protocol as the security agent. The heavy weight of DTLS protocol used to protect the communication between smart objects on IoT. On some of the area, DTLS may not secure for reliable data transferred and can be considered as the threat for the protocol. DTLS do not supporting for multicast messages in communications on IoT. Having a lack of security in DTLS protocol, Random session key generation stream algorithm is used to protect the communication among the object's security. The various implementation of CoAP protocols on IoT may lead towards the secure communications among smart objects.

In future work we implement multicast communication then design cryptography algorithm then test or implement algorithm in cooja simulator and analyze result.

8. Reference

1. Sheng Z, Yang S, Yu YA. Vasilakos V, Mccann JA, Leung KK. A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities. IEEE Wireless Communications. 2013; 20(6):91-98.
2. Qi Jing, Athanasios V. Vasilakos, Jiafu Wan, Jingwei Lu, Dechao Qiu. Security of the internet of things: Perspectives and challenges. Wirel. Netw. 2014; 20(8):2481-2501.
3. Wan J, Tang S, Shu Z, Li D, Wang S, Imran M, Vasilakos AV. Software-defined industrial internet of things in the context of industry 4.0. IEEE Sensors Journal. 2016; 16(20):7373-7380.
4. Kraijak S, Tuwanut P. A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends. In: 11th International Conference on Wireless, 2013.
5. Communications, Networking and Mobile Computing (WiCOM 2015). 2015, 1-6. DOI: 10.1049/cp.2015.0714.
6. Administration. Internet of Things. (accessed November 3, 2016. 2016 URL: https://en.wikipedia.org/wiki/Internetofthings
7. Xi Chen. Constrained Application Protocol for Internet of Things. URL: https://www.cse.wustl.edu/~jain/cse574-14/ftp/coap/
8. Khattak A, Ruta M, Sciascio ED, Sciascio D. Coap-based healthcaresensor networks: A survey, in Proceedings of 11th International Bhurban Conference on Applied Sciences Technology (IBCAST) Islamabad, Pakistan. 2014; 499-503.
9. Ugrenovic D, Gardasevic G. Coap protocol for webbased monitoringin iot healthcare applications, in 23rd Telecommunications Forum Telfor(TELFOR). 2015, 79-82.
10. Wei Yu, Jangwon Lee. Dsr-based energy-aware routing protocols in ad hoc networks. In Proc. of the International Conference on Wireless Networks, 2012.
11. Zhao P, Yang X, Yu W, Fu XC. A loose-virtual-clustering based routing for power heterogeneous manets. IEEE Transactions on Vehicular Technology. 2013; 62(5):2290-2302.
12. Huang Y, Yang X, Yang S, Yu W, Fu X. A cross-layer approach handling link asymmetry for wireless mesh access networks. IEEE Transactions on Vehicular Technology. 2011; 60(3):1045-1058.
13. Bellavista P, Cardone G, Corradi A, Foschini L. Convergence of manet and wsn in iot urban scenarios. IEEE Sensor Journal. 2013; 13(10):3558–3567.
14. Patel DN, Patel SB, Kothadiya HR, Jethwa PD, Jhaverii RH. A survey of reactive routing protocols in manet. In Proc. of IEEE International Conference on Information Communication and Embedded Systems (ICICES), 2014.
15. Alani MM. Manet security: A survey. In Proc. of 2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2014.
16. Chawda K, Gorana D. A survey of energy efficient routing protocol in manet. In Proc. of 2nd International

Conference on Electronics and Communication Systems (ICECS), 2015.
17. Goncalves P. An evaluation of network management protocols. In Proc. of IFIP/IEEE International Symposium on Integrated Network Management, 2009.
18. Yu Y, Ni L, Zheng Y. Survey of qos multicast routing protocols in manets. In Proc. of 2011 International Conference on Computer Science and Service System (CSSS), 2011.
19. Herberg U, Clausen T, Jacquet P, Dearlove C. The optimized link state routing protocol version 2, 2014. Internet Engineering Task Force, RFC 7181
20. Qi Jing, Athanasios V Vasilakos, Jiafu Wan, Jingwei Lu, Dechao Qiu. Security of the internet of things: Perspectives and challenges. Wirel. Netw. 2014; 20(8):2481-2501.
21. Wan J, Tang S, Shu Z, Li D, Wang S, Imran M, Vasilakos AV. Software-defined industrial internet of things in the context of industry 4.0. IEEE Sensors Journal. 2016; 16(20):7373-7380.