WoT-UAF: User Authorization Framework of the Web of Things

Kaptan Singh, Deepak Singh Tomar

Abstract— The Web of Things (WoT) promises to dramatically boost the potentiality of interconnecting smart and physical devices over the Internet. In this paper, token based secure framework for Web of Things devices are discovered. To assure secure authenticate devices from unauthorized client, the framework is based on OAuth 2.0 framework. This framework reduces the need to share credential of the client with the connected devices, they overcome the weakness of the traditional sever client model. It provides security between Web of Things client and WoT devices communication.

Keywords: Web of Things, Authorization, MQTT, OAuth 2.0, Secure Framework.

I. INTRODUCTION

WoT is the Internet of things or everything is also Internet changing idea of multiple object related with smart network and interaction along one another and with humans. In WoT, information transfer with self and cloud is exchanged, hoping about accuracy in gathering, taping along with analyzing currents data [1]. In addition, web-enabled items should be used again and accept established web mechanisms such as search, browsing, linking and caching, as [2].

The Internet of things and web things conceive both idea of conversation being wide, anywhere plus everywhere. The ideas cannot be approved in existence without giving priority to privacy and Security. Optimizing web technologies provides an intangible complication of low-level protocols.

For example, HTTP and WebSocket is used again through smart things. In addition, developer develop process that communicate via smart things along one another [3, 4]. An open problem in this area, is the smart and allowance of customers to have access to the facilities due to the dangers like:

-Malicious customers and unwanted data sharing
- Anytime and anywhere
- Unexpected work load and availability risk

Unknowingly, the concept are strong for below aspect. Firstly, asymmetry: IOT is composed of infinite dissimilar objects through different platforms, protocols and needs. Secondly, lack of resources: reason being the demand procedure. Third, identity and authentication: Traditionally, identifiers linked to users to examine if someone was permitted to take action on issue. Combination of hardware and technical world where things are smart of working on oneself or someone else's work.

OAuth 2.0 is an authorization framework [5]. It was designed to allow a third party application to access the restricted resource under the control of users, without sharing users credential. The architecture holds four factors that are third party application willing to access to protected resources (i.e. the client). The Resource Owner able to permit or not permit to a resource data which is protected. Server of the resource that exposes the resource which is protected and the server which authorizes control the further process. This factor establish and End-toned transport layer security (TLS) channel & communicate among the resources which giving the access procedure. Which is summaries as below:

1. The client connect with the resource access procedure.
2. The owner of the resource grants the access to its client by transporting a code which is authorized.
3. The client sends the received authorized code to server.
4. The authorized server matches the code of authorization and release a token which contains the detail of the provided control to reach out to the client.
5. The client forwarded the token to the server of resources.
6. Then the resource server validate the token received is supportive case, the definition of the resource token structure is beyond the scope of the standard.

Any standard definition cannot define the token structure. For directly controlling the access to all resources OAuth 2.0 cannot be directly implemented within a defined framework of Internet of Things. Computational and bandwidth capabilities of the devices with contained resource are not very much, as a result it does not support the installation of the connection message. Apparently, the IETE [14] ACT working group is forwarding and encouraging the adoption of a light version of implementation of OAuth 2.0. Where both resource and application are available within the defined Internet of Things network commonly to both. OAuth 2.0 is Installed on constrained device where its interaction are authorized by mean of CoAP (Constant application Protocol) message, which arrive through a datagram transport layer security (DTLS) [13]

II. PRELIMINARY BACKGROUND

The web of things [6] aims to promote the ability for computers to exchange and make up the use of information between IOT devices. It does this by defining METADATA format. WOT can be described with wide range of IOT

Revised Manuscript Received on September 10, 2019.

Kaptan Singh, Maulana Azad National Institute of Technology, Bhopal, Madhya Pradesh, India
Deepak Singh Tomar, Maulana Azad National Institute of Technology, Bhopal, Madhya Pradesh, India

network interface. A thing description can describe the network interface of existing devices. Which can be provide and used by the device running a run time of WoT supporting script APT of WoT. An API that formulates the inter device interaction with a common abstraction layer it also supports attaching additional information related to reader's reference based on the linked data [7] supporting powerful search capabilities.

The- web of things (WoT) architecture [6] defines three basic activities that can be organized into various configuration and topologies based on the concrete deployment scenarios:

- **WOT thing** – An entity of the software which represents virtual and physical IOT devices and exposes an interaction with network facing API for interaction. Each WoT has an attached description thing (TD) [8]. A (TD) encrypts and information set about things like same categorization it is available interactions and communication with security mechanisms. Traditionally a WoT thing portrays the role of network that responds but doesn't start the interactions. For example – WoT things may be a controller of the garage, this can have multiple interactions that can be performed by another WoT thing on garage door like open, close etc.
- **WOT Client**- an entity can be operated on WOT thing. It is able to consume TD provided by another WOT thing and en-woke interactions on network interfaces. For example – A WOT client might browse on the smartphone of the user which allows users to induce one of the provided interactions by the garage door controller.
- **WOT Servient**- an entity that can be visible as a combination of client and server. A sever provides one or more WOT things interface as a [server] and at the same time. It is able to operate on WOT client. For example-a WoT Servient may be executing a service on the gateway device providing a general lookup. Service with its own network interface. The typical architecture of a WoT Servient is shown in Fig 1. A WOT Servient supports one or more thing description and associated protocol bindings templates. The templates of binding are used to showcase td for a specific IoT protocol such as OFC [open community foundation], HTTP [hypertext transfer protocol], and COAP [constrained application protocol][20 AND MQFT. A WOT Servient supports can also take a WoT runtime a scripting API of WoT. The scripting API is a voluntary constituent that allows to implement the application logic. Servient is a standard way of using a higher level performing language. However in this data we lay emphasis on data security of the data metadata alone and not consider the other security implication of the scripting API.

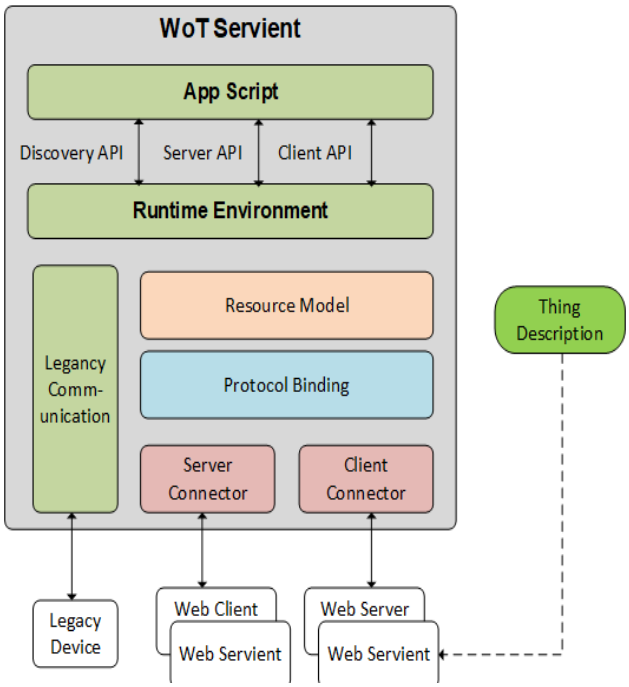

Fig. 1 WoT Servient architecture

Thread mode was due to the lager diversity of the devices, use cases, development scenarios and requirement of WOT things. However it's impossible to describe a single model of threat of WoT moved fit in every use cases, so we have created a complete threat in WoT framework [9] which is adjustable by OESM [original equipment manufacturer] or other WOT system users based on the exact security requirement. The thread model defines security relevant WOT and a set of threads on these assists.

2.1 OAuth 2.0

OAuth 2.0 is a token based authentication and Authorization open platform standards communication for internet defined in [5]. Traditionally in the client server communication model, protected resource on the server are accessed by the client using the credentials of the user.

If there is a third party client included, the owner of resource secret key need to be shared with the client. OAuth 2.0 will provide a framework for taking decision on authentication of data and also protect it by adding a layer of authorization.

Instead of using the owners secret key of the resource the client is directly provided access to resource which is provided by granting the client the token of access by authorization server. The token of access define the scope, lifetime and other accessible attribute. Following 4 entities are involved in this- a resource owner (RO), Authorization Server (AS), A Client, Resource Server Host.

The flow of information in OAuth 2.0 is described as follows- the request of Authorization is sent by the client to Authorization Server. The Authorization Server AS identifies and authenticate the client and reply with an access token. This access token is used by the client to authenticate itself to the server of the resource and access it. OAuth defines four roles:

Resource Owner- Resource Owner can grant access to resource which is protected.

A resource Server- it hosts the resource which is protected. It also can reply and accept the message to a resource and from a resource which protected using access token.

Client- An application making resource which is heavily secured.

Authorization Server (AS)-this server issues token to the client after it passes through the Authorization layer.

2.2 IETF token formats

The format of token are out of scope of OAuth 2.0. However there are some interesting proposals coming up from IETE regarding it. For eg. Bearer Tokens [10] are simply defined as nothing but contained of data or information. The major part of OAuth 2.0 implementation work of this solution even after security mechanism not being present. And as a result the confidentiality and security of the token is challenged to the compulsory TLS Protocols. The JSON web Token (JWT) is a compact way for carrying access grant. It average the find JSON format and claim including time, validity, issues of the owner and time of revolution also the field of authentications.

The possibility of expending the token format with claim private and unregistered when out to be useful for particular specific purpose. And finally the token of Proof-of-Possession [12] (POP) can be used in area where extra security is demanded. For instance thoughtfully, POP tokens is nothing but a scenario where a client should illustrate the ownership of encrypted key whereas making demand for getting to secured resources. Both symmetric and asymmetric can be used in this case.

Security Requirements the security objectives of the proposed protocol are:

- There should be mutual authentication between IOT devices on a resources server using an access token obtained from the authorization server.
- Limited access should be gains to another Internet of Things Device.
- Security levels should be according to the need of the application
- Occurrence of conflict in the Internet of Things (IOT) device, should not give access to the user to handle the security of scheme.

III. PROPOSED FRAMEWORK

The reference architecture showed in fig. 2

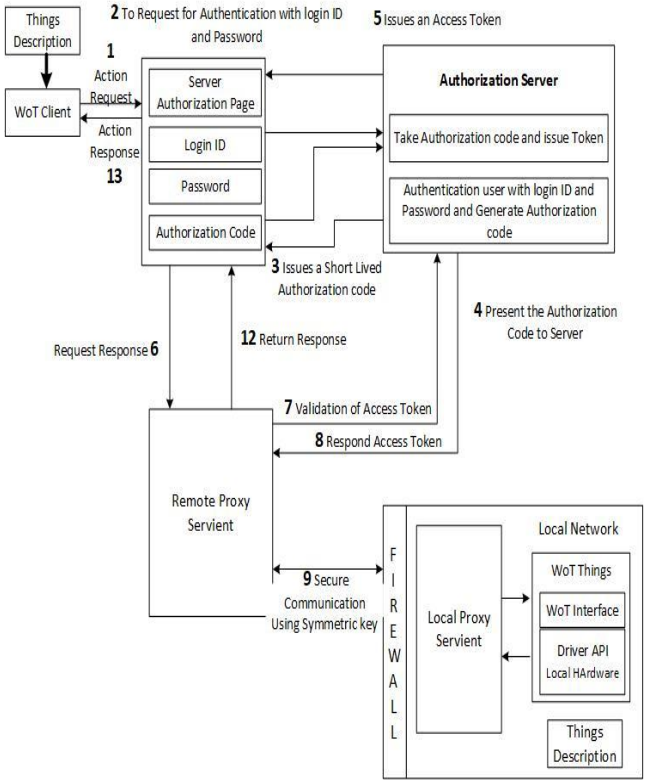

Fig 2: proposed framework for Web of Things

WoT Servient provide control, access status from web of things devices. The functionality of each module is follows:

Web Server: web server take request from WoT client through network and sends to response to WoT client. HTTP, CoAP, MQTT protocol are used between Servient and client.

Web Client: Basically Web Server can communicate other Web Server and Servient via network. Web Client communicate with other server.

Legacy Communication: Smart Devices like IoT devices are directly communicate with each other or Servient. But Outdated devices which are not smart device but present in network. In this case ligancy communication module implemented for this tip of devices, so they communicate with smart devices.

Protocol Mapping: Protocol Binding Templates comprise of reusable lexicon and plan design expansions to the WoT Thing Description organize that empower an application client to associated, employing a steady interaction demonstrate, with Things that uncover different conventions and convention utilization.

Resource manager: WoT Servient manage as a Server for client request, manage as a client for other server request, manage as controller for inside status of the Things, all things make a management block called Resource manager.

Things Description: Things Description define the competences and properties of things or a legacy Devices.

Server API use for making server functions, Client APT use for making Client Functions and App Script creating Application API and call above API.

Secure protocols should be used to protect authenticity and confidentiality and provide replay protection when data is exchanged between the WoT Client and the Remote Gateway Servient. If the WoT Client have Access Token it means they are authorized. After Authorization Remote Gateway Servient take the message and packed as a WoT request package and transfer to the Local Gateway Servient by secure Channel. The communication between Local Gateway Servient and WoT Things can be unprotected because local network work as closed nature, but it is recommended to use suitable security mechanism additionally.

In the OAuth 2.0 authorization framework, every WoT client needs to authenticate itself with Authorization server for access WoT device or Resource Server. The Authorization Server provide Access Token to WoT client with lifetime of session. There are two phases in this work: first token exchange and Authentication and second is Authorization.

The authorization server start with one time password for each client. The Time based One-time Password algorithm can be used for this. Once the Client verified with one time password than authorization server issue access token. After exchange access token client communicate WoT device Directly without Authentication server. But access token has a time interval or session time if token will not use in long time than token can be expire and Client demand to server for refresh token.

3.1 Authentication and Token Exchange:

1. There is the client (web of Things client) that would like to access the WoT device those are in IoT networks. The WoT client redirect to Service Authorization page.
2. The WoT (web of Things) Client Authorization pages take Login Id and Password from WoT client and encrypt with elliptical curve cryptography (ECC) algorithms. Suppose a message 'M' have client ID, Login ID and Password and encrypted with ECC algorithm. Consider *'M'* has the point *'m'* on the curve *'E'*. Randomly select 'k' from [1 – (n-1)]. Two cypher text are generated C1 and C2 and sends to Authorization Server. Authorization Server decrypt this message with private key and retrieve original message.
3. The Authorization Server (OAuth 2.0) search this credential in memory to locate WID. If WID is not found than authentication request is rejected and sent error massage. Otherwise Authorization server generate the Short time code (STC) for WoT Client.
4. WOT client receive this code and give response to authorization server in give time interval. If Authorization server receive response in time than generate Access Token (AT) with following fields.

WID_A: identity of the WoT client requesting access to WoT device.

AO: identity of Authorization Server who issued the token.

T_i: time to live field when the token expires.

Scope: the scope field provide access control.

The Access Token is encrypted with ECC algorithms and generate encrypted message M = {WID_A, AO, T_i, Scope} to WoT client. To ensure the data integrity, the Authorization Server sent MAC address to WoT client. The WoT client receive the Access Token AT and verifies the Token. If verification fails the authentication request is terminated. Otherwise go to proceed second phase Authorization.

3.2 Authorization Phase

After Authentication and Access Token validation next phase will be stared. In this phase WoT client access WoT device using Access Token. The following Steps are required for Authorization Phase show in fig 3:

1. WoT client receive this message, decrypt with private keys and take Access Token. The WoT client make a Request message M, encrypt with symmetric key and send to RPS with corresponding MAC address.
2. RPS checking the message if they found Access Token in message than firstly they check the authencity of Access Token. RPS sends this Access Token to Authorization Server for validation. Authorization server search this access token in memory. If not found than error message generated otherwise send validation message to RPS. After validation of Access token RPS make encryption on message and encrypted message send to Local Proxy Server (LPS).
3. Local Proxy Servient enable with proxy firewall. Firewall decrypt this message with symmetric key and obtain WoT Device ID. WoT Things check the WoT device MAC if found wrong than rejected the request otherwise fulfil the request of WoT client.

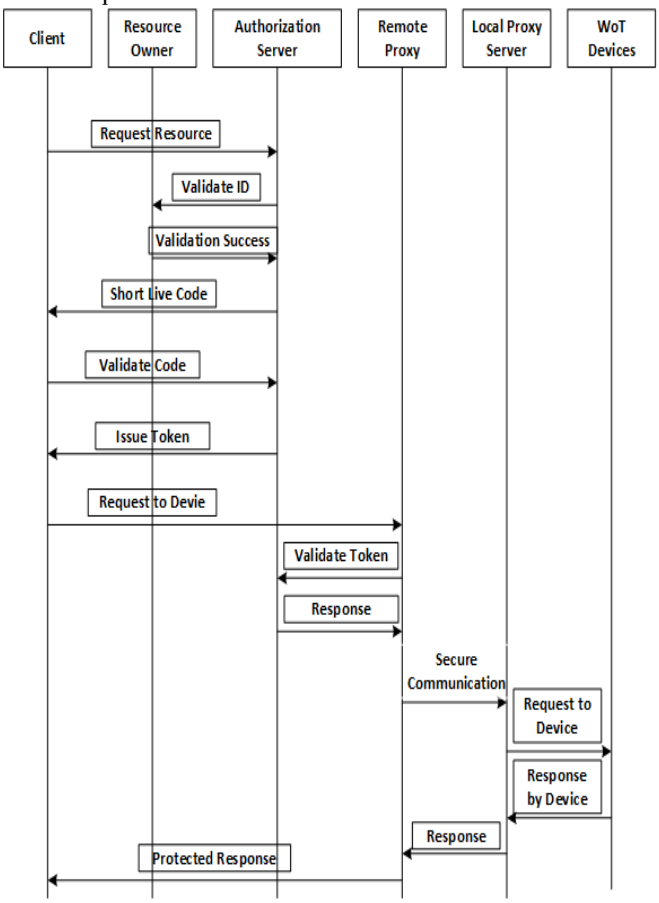

Fig. 3 Authorization and Authentication Phase

IV. RELATED WORK & RESULTS

MQTT based IoT model depend upon user authentication of user and deices. A user Authenticate with user application ID and Application Secret and Devices identified by pair of credential of Device ID and Device Secret. This mechanism implemented on MQTT based IoT service Platform [15]. He takes three types of devices arduino mega 2566, Raspberry Pi 2 model B and Chrome browser. All three deices connected authserver ia HTTP. The average delay during user authentication with Authserver using known credential and issuing unique devices ID and Devices Secret is 121ms. The average delays to obtain both Token with Arduino mega 2560 being the slowest shown in the fig 4. the Average Request token delay and the average access token delay of all cities are 1209ms and 755ms because all end user browser located in different cities and the standard deviation are 427ms and 361ms shown in fig 5.

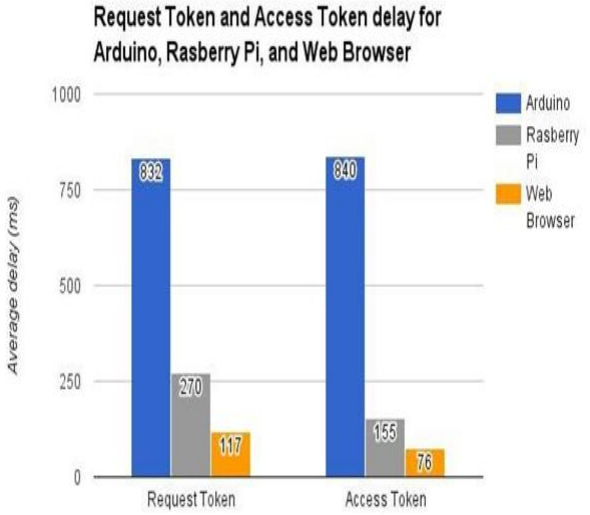

Fig. 4 Token request delay for different device platforms.

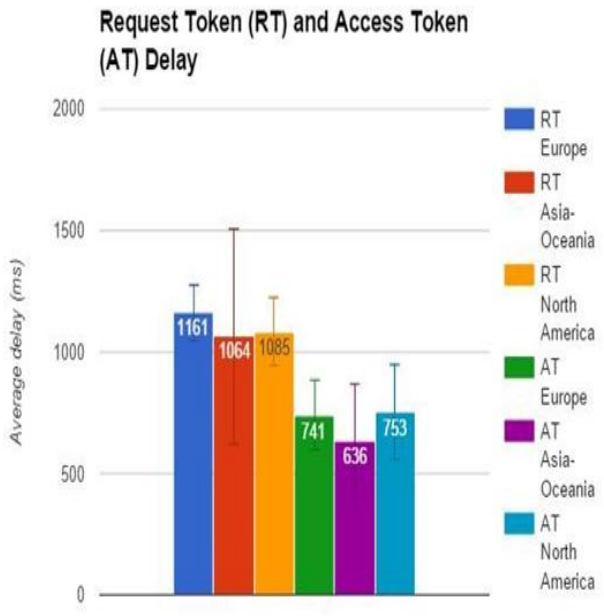

Fig 5 Token request delay experienced in different parts of the world.

Powertrace[16] are used for assessment of the energy consumption in CoAP. The Powertrace evaluated energy consumption of each component on the board. To determine energy consumption MSP430f5438 microcontroller, T1 cc2420 radio chip and exp5438 mote are used in Cooja simulator[21] in Contiki OS. Fig 3 show the aggregate energy consumption calculation for deferent protocol executed on Contiki system. In Contiki system, there are two model for, one is work model and another is suspended model known as LPM (low power model). The two model of radio chip is RX for listening and TX for the radio chip sending message. The fig 6 show the CPU active mode consumed most energy in process. As for LPM, CPU is really busy when process the request so it is not must time left for low power node because of only one packet send at a time so the TX mode does not consume must energy. The energy consumption of other protocol are provided in the thesis about IoT-OAS [17][18].

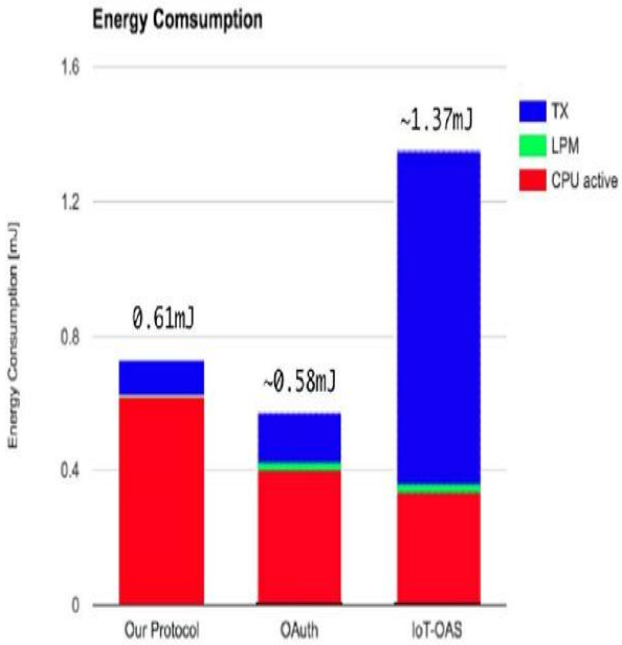

Fig. 6 Aggregate energy consumption

M. Vucinic, B. Tourancheau, F. Rousseau, A. Duda et at [19] proposed Object security architecture for Internet of Things. OSCAR evaluated three aspects: Overhead elliptic Curve digital Signature Algorithm on nodes, second is Scalability in machine to machine communication and the last one is impact of the radio duty cycling mechanism. Fig 7 show the overhead of the ECDSA computation on the WiSMote[22] and ST GreenNet[23] Platform. The use of 32 bit Micro controller Unit reduce the computational time by a factor of 4, at that point energy consumption reduced by factor of 3.084. Advancement of hardware technology reduce the energy consumption and computation cost for low power MCU devices.

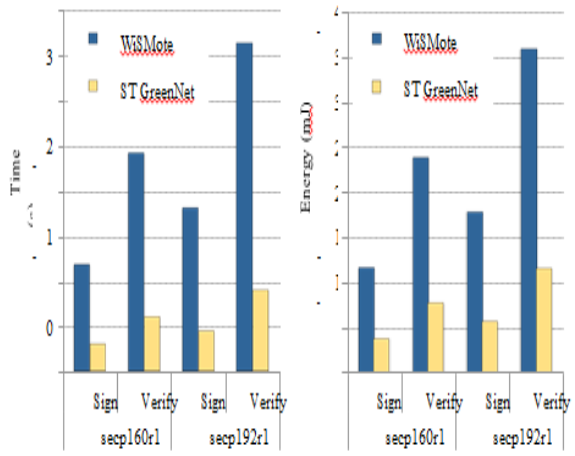

Fig 7: (a) ECDSA computation and energy benchmarks at 21.3 MHz for 16-bit (WiSMote) and 32-bit (ST GreenNet) hardware platforms.

Resource signing load at the producer is an important aspect of performance evaluation. Scalability as a function of the radio between the total number of DTLS client or consumer and the maximum number of open session at a DTLS server. Mean resigning interval beta(B) = t/N where N is the total number of secured resource on the producer. Fig 8 show the impact of the traffic generated by OSCAR on energy consumption. When the client/session slot ratio is approximately 1.3 OSCAR cross the energy performance of compressed DTLS in the case of WiSMote. But in the case of GreenNet this client/session slot ratio increased approximately 2.15.

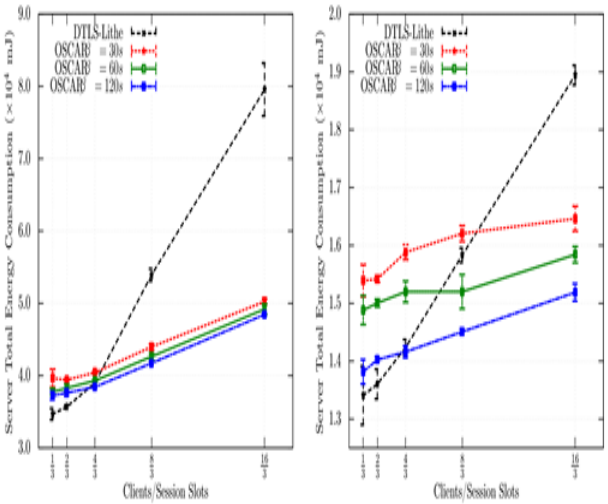

Fig 8: Constrained server total energy consumption over 3 hours.

The goal is to relieve constrained producer fro, traffic and place burden on Consumers. The ECDSA verification result much better the compressed DTLS approach whenever the client/session slot ration is 3.7 and 4.17 notice in fig 9. Suppose 16 clients are present in the network, a node running the DTLS client on WiSmote spend 75.1% of energy on MCU computation and the same scenario OSCAR spend 83.2% of energy on MCU computation which is 8.1% more than DTLS. If use Green Net platform DTLS use 79.7% of energy on MCU computation and w90.1% with OSCAR. Finally evaluate the request response latency in fig 10.

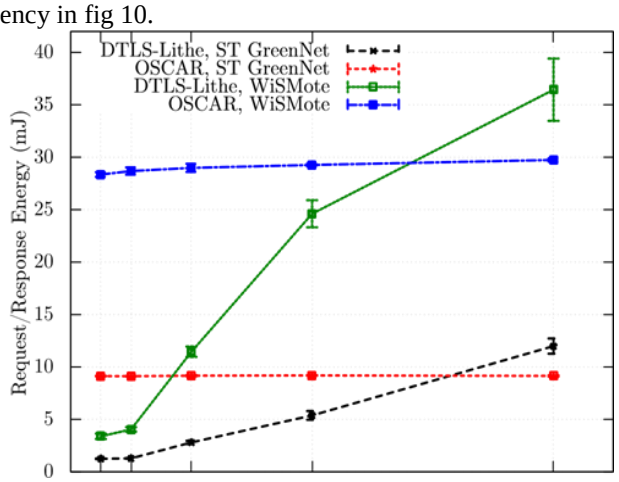

Fig 9: Client energy consumption per CoAP request-response. It includes a possible DTLS handshake

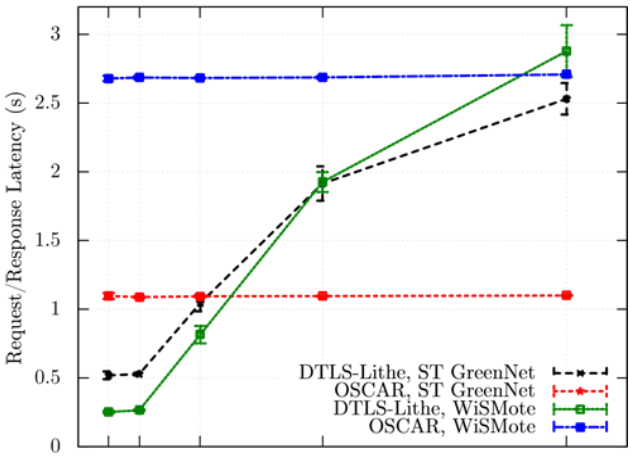

Fig 10: Request-response latency. It includes a possible DTLS handshake.

V. CONCLUSION

This paper present the secure framework for web of Things. The proposed framework is based of OAuth 2.0 framework. The propose framework is based on JSON access Token. It provides security between Web of Things client and WoT devices communication. The following functionalities of the secure framework is: (i) it control access request prepared by third party application through OAuth 2.0 authorization framework. (ii) It support JSON token format for Authentication and Authorization. Component of the WoT framework have been defined. Authorized WoT client access WoT devices as well as legacy Devices securely.

VI. REFERENCES

1. J. Höller, V. Tsiatsis, C. Mulligan, S. Karnouskos, S. Avesand, and D. Boyle, From Machine-To-Machine to the Internet of Things. Elsevier, pp. 233–235, 2014.
2. Framework of the web of things, ITU-T Y.2063, 2012
3. Guinard D, Trifa V, Wilde E. A resource oriented architecture for the web of Things. In:Internet of things (IOT), Tokyo, Japan. Nov 2010. p. 1–8.

4. Guinard D, Trifa V. Towards the web of things: web mashups for embedded devices. In: Workshop on mashups, enterprise mashups and lightweight composition on the web. Proceedings of WWW (international world wide web conferences). 2009. p. 15.
5. "The OAuth 2.0 Authorization Framework", IETF RFC 6749, 2012.
6. K. Kajimoto, U. Davuluru, and M. Kovatsch, "Web of Things (WoT) Architecture," W3C, W3C Working Draft, Sep. 2017. [Online]. Available: https://www.w3.org/TR/2017/WD-wot-architecture-20170914/
7. T. Heath and C. Bizer, Linked Data: Evolving the Web into a Global Data Space, 1st ed., ser. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, 2011, vol. 1:1. [Online]. Available: http://linkeddatabook.com/editions/1.0/
8. S. K¨abisch and T. Kamiya, "Web of Things (WoT) Thing Description," W3C, W3C Working Draft, Sep. 2017. [Online]. Available: https: //www.w3.org/TR/2017/WD-wot-thing-description-20170914/
9. E. Reshetova and M. McCool, "Web of Things (WoT) Security and Privacy Considerations," W3C, W3C Note, Sep. 2017. [Online]. Available: https://www.w3.org/TR/2017/WD-wot-security-20171116/
10. Ashutosh Dubey and Shishir K. Shandilya, "Exploiting Need Of Data Mining Services in Mobile Computing Environments", Computational Intelligence and Communication Networks (CICN), 2010
11. R. Chaure, Shishir K. Shandilya, "Firewall anomalies detection and removal techniques – A survey", International Journal of Emerging Technologies, Vol. 1(1), pp. 71–74, 2010
12. A.K. Dubey, Shishir K. Shandilya, "A comprehensive survey of grid computing mechanism in J2ME for effective mobile computing techniques," Industrial and Information Systems (ICIIS), pp.207-212, 2010
13. Shishir K. Shandilya, S. Jain, "Opinion Extraction & Classification of Reviews from Web Documents", Advance Computing Conference IEEE International, 2009.
14. Smita Shandilya, SK Shandilya, Tripta Thakur, Atulya K Nagar, Handbook of Research on Emerging Technologies for Electrical Power Planning, Analysis, and Optimization, 2016
15. Shishir K. Shandilya, Smita Shandilya, Kusum Deep, Atulya K. Nagar, Handbook of Research on Soft Computing and Nature-Inspired Algorithms, 2017
16. Patel, A., Jain, S., Shandilya, S.K., Data of semantic web as unit of knowledge, Journal of Web Engineering, 2019
17. Shandilya, S.K., Shandilya, S., Deep, K., Nagar, A.K., Handbook of research on soft computing and nature-inspired algorithms, IGI Global, USA, 2017
18. Shandilya, S., Shandilya, S.K., Thakur, T., Prioritization of Transmission Lines in Expansion Planning Using Data Mining Techniques, Lecture Notes in Electrical Engineering, 2019
19. Shandilya, S.K., Ae Chun, S., Shandilya, S., Weippl, E., Internet of things security: Fundamentals, techniques and applications, 2018
20. Shandilya, S.K., Ae Chun, S., Shandilya, S., Weippl, E., IoT security: An introduction, River Publishers, Denmark, 2018
21. Netpie.io, 'NETPIE | Network Platform for Internet of Everything', 2015. [Online]. Available: https://netpie.io. [Accessed: 04- Dec- 2015].
22. A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, "Powertrace : Network-level power profiling for low-power wireless networks lowpower wireless," Swedish Institute of Computer Science, 2011.
23. T. Instruments, "Cc2420 datasheet," Reference SWRS041B, 2007.
24. T. Instruments, "Msp430f5438 datasheet," Reference SLAS655B, 2010.
25. M. Vucinic, B. Tourancheau, F. Rousseau, A.Duda, L Damon, R. Guizzetti, "OSCAR: Object Security Architecture for Internet of Things" Ad Hoc Network, jan 2015
26. Z. Shelby, K. Hartke, and C. Bormann, "Constrained Application Protocol (CoAP) draft-ietf-core-coap-18," IETF work in progress, 2013.
27. M. Vu˘cini´c, B. Tourancheau, F. Rousseau, A. Duda, L. Damon, and R. Guizzetti, "Energy Cost of Security in an Energy-Harvested IEEE 802.15.4 Wireless Sensor Network," in MECO. IEEE, 2014.
28. K. Hartke, "Practical Issues with Datagram Transport Layer Security in Constrained Environments draft-hartke-dice-practical-issues-00," IETF work in progress, 2013.
29. M. Vu˘cini´c, G. Romaniello, L. Guelorget, B. Tourancheau, F. Rousseau, O. Alphand, A. Duda, and L. Damon, "Topology Construction in RPL Networks over Beacon-Enabled 802.15.4," in ISCC. IEEE, 2014.