An Overview Of Iot Operating System: Contiki Os And Its Communication Models

Payal Malik, Malvika Gupta

Abstract: The virtually dependent such as internet and computers on the human for information that captured the data in form of typing, pressing button, digital picture or scanning a bar code with unevenly of 1024 terabyte and accessible on the internet. In this article author has described an IoT based operating system CONTIKI OS used for transmitting the data from one to another node with authenticity. CONTIKI OS interface is based on the JAVA which is efficient and secure by many security layers. Protothread, system loop and micro ip are the features that has been supported by the CONTIKI OS and also supports over network such as TCP/IP and IPv6 stack by cisco. Author also discussed about the communication models that helps to transmit the data over network with secure connection. This conclude that CONTIKI OS provides the security while transmitting the data from one to another node using communication models.

Keywords: IoT, IoT Operating System, Contiki OS, IoT models.

1. INTRODUCTION

The interconnection of objects that profound currently and the objects not only gather the information from sensors with a interaction of physical world while delivering the service to transfer information and application [1]. The innovation technology with an open source Contiki OS that make the implementation efficient with a memory of elementary protothread. The multithreading model is compared as same as stack in which the less computation and memory is required in concurrent processes. The protothread is also supported by the Contiki OS where the event driven, and explicit manner is considered to write the codes or programs which tough to understand and maintain. The model of execution is driven by the mixing of stackless thread with a linear event. The Cooja simulator is being used under a Contiki OS that work on the JAVA interface. The advantage of Contiki OS is power profiling and low power radio networking for the network sensors with efficient optimization. It focuses on TCP, ICMP and IP protocols [38] with a minimum absolute feature if full TCP/IP stack and the limited resources devices of TCP/IP and the sensors nodes. IoT models involve the networking of smart objects (RFC) techniques for transmitting the data/information over network. Communication models for IoT such as Device-to-device in which small data packet is being used for the information transmission between devices at low rate requirement in this model mostly such as home automation systems. Device-to-cloud this enables the enhancement to the end-user demands as the capability of product bond extended. Device-to-gateway in which connection is being established between the devices to application layer and it helps to have transaction between the devices to cloud services. Back-end-data sharing which consist of hybridization of aforesaid two models device-to-gateway and device-to-cloud that helps to transmit the data in fast and efficient manner.

2. OPERATING SYSTEM

The theoretical research in the area of WSN such as sensors network that modelled the lifetime into real application development which was conducted by "Swedish institute of computer science" and it is contributed by SICS (Adam Dunkels) currently CEO of Thinksquare (focus on IoT ideas and innovation through interconnectivity of devices) basically work on the Contiki OS that is a lightweight and portable constrained environment. The TCP/IP and pre-emptive multithreading is an intersecting feature that is supported by the Contiki OS with an implementation on IPv6 stack by Cisco and Atmel [31]. The innovation technology with an open source Contiki OS that make the implementation efficient with a memory of elementary protothread. The model of execution is driven by the mixing of stackless thread with a linear event [32] and the advantage of Contiki OS is power profiling [33] and low power radio networking for the network sensors with efficient optimization. Basically, it consists of two division: -

- Core
- Loaded Programs

The division is basically work on the time of compilation while deploying, the core consists the Kernel of Contiki and the run-time language with libraries is consist by Loaded Programs that help to communicate the devices driver with hardware's.

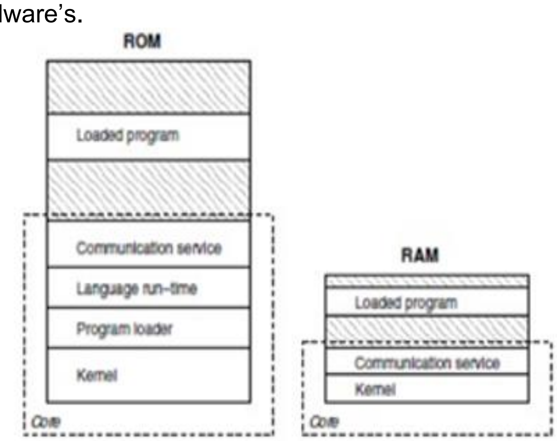

Fig.1. *Contiki Operating System Partitioning*

The backend language of Contiki OS is interfaced by JAVA and in a Cooja simulator tool the C language is used that make easy to switch the various platform and approx. 20 platforms is released since 2003[[34].

- *Payal Malik 1, Malvika Gupta 2*
- *1 ABES Engineering College, Ghaziabad,UP,India*
- *2 ABES Engineering College, Ghaziabad, UP, India*

2.1. Various Contiki OS features in IoT:

Event based Kernel
The execution or implementation of code is done by event handler that means the code is fully depends on the event and never be interrupted by the single code block. The multithreading model is compared as same as stack in which the less computation and memory is required in concurrent processes [35].

Protothread
Apart from the pre-emptive thread the protothread is also supported by the Contiki OS where the event driven, and explicit manner is considered to write the codes or programs which tough to understand and maintain. It persistence the high-level function implementation with abstraction of programming language and without any overhead threads the conditional blocking is performed. 2 byte of RAM each [34] [35] is required for single protothread with stackless in Contiki OS.

System Loop
To initialize the few process, it repeatedly uses a function process_run () [35] for the start-up of Contiki OS. This function is registered by poll handler where the processes on event from event queue system after this the function is call back by the handler that is still in the queue. While it will go to sleep if the event queue is exempted, an external interrupt is required to awakes by the function process_run () with new hander event.

```
Int
Main(void)
{
Beep();
While(1)
{
While(process_run()>0)
/*watchdog_reset();*/
Lpm_sleep();
}
Return ();
}
```

Above source code is taken from Contiki and tells the common demonstrate of usage of process_run () [37] and hardware platform. Whenever the node wakeup again the Lpm_sleep() is called thus we use an external wakeup Lpm_awake().

Micro IP (uIP)
It focuses on TCP, ICMP and IP protocols [34] with a minimum absolute feature if full TCP/IP stack and the limited resources devices of TCP/IP and the sensors nodes. To contain one maximum packet size in single global buffer that is enough large to hold the packet. To notify the uIP application of the data packet analysis and network devices when the new uIP is call to handle the data. The data that has to be overwritten on the packet by another incoming packet to avoid we uses uIP application because it has only single buffer.

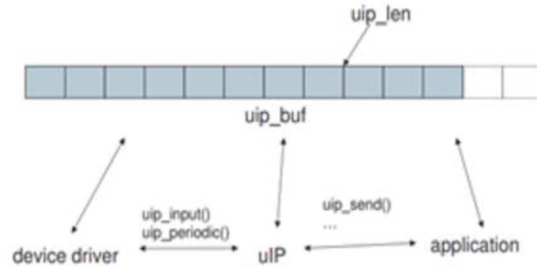

***Fig.2**. uIP's Global Buffer Usage*

The current data size is stored in the integer value variable uip_len and for global buffer we use uip_buf [33]. Functions assign for device drivers are: -

- Uip_input(): The packet received by the device driver and stored in the global buffer while it is checked by device driver for the no outbound packet.
- Uip_periodic(): For any retransmission is needed we this function periodically through device driver.
  The Cooja simulator is being used under a Contiki OS that work on the JAVA interface and in the Cooja tool we use a C language as a backend for programming. Various OS is being used rather Contiki OS:
- mbed OS: It uses a C++ framework is used for the development in IoT application and run over the less capacity, memory, storage and processing is been designed [32].
- TinyOS: For non-commercial OS designed for wireless sensor network and running resource constraints with feature of management network and ubiquitous computing [33].
- MicroC OS: Designed for real-time OS and embedded devices for low power microcontroller [34].
- RIOT OS: it usages C++ framework with efficient high degree of programming that make optimize utilization of microcontroller and based on microkernels [34].
- Brillo OS: New OS for low power devices introduced by Google in 2015 such as android platforms (android smartphone) [35].

OS	ContikiOS	TinyOS	RIOT OS
Minimum RAM	Less than 2kb	Less than 1kb	Approx. 1.5kb
Minimum ROM	Less than 30kb	Less than 4kb	Approx. 5kb
C	Fractional	No	Yes
C++	No	No	Yes
Multi-threading	Fractional	Fractional	Yes
Real-time	Fractional	No	Yes
Modularity	Fractional	No	Yes

***Table.**1. Operating System Requirement*

3. COMMUNICATION MODEL IN IoT

The networking of smart objects (RFC) [22], a guiding was released in March 2015 by the Internet Architecture Board. Four frameworks for communication model are: -

Device-to-Device Communication

The direct connection is established between two or more devices and communicate with each other rather than immediate application server use. Several networks are using to communicate with several protocols used in it such as Bluetooth [23], Z-wave [24], or ZigBee [25] is established to direct device-to-device communication. The small data packet is being used for the information transmission between devices at low rate requirement in this model mostly such as home automation systems. Many interoperability challenge approaches are proven by the device-to-device-communication as per the IETF journal article "The direct relationship is often between the devices, the built-in mechanisms of privacy and trust they usually have but the data modes usually uses device-specific that require redundant development effort" [26]. The device specific data format is being implemented rather than open approach that uses standard format.

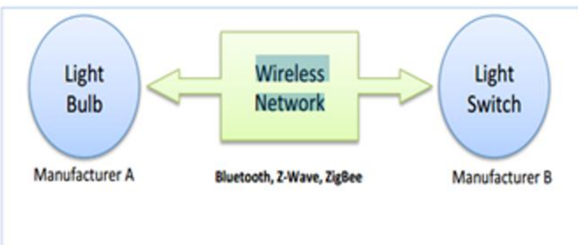

Fig.3. *Device-to-device communication*

Device-to-Cloud Communication

There is a direct connection is established between the devices to the internet cloud that helps in exchanging information and traffic control. The connection is established between IoT devices to IP network that lead to cloud services this is done or uses with tradition wired or wireless network. "Smart TV" [27] is the most popular example for the IoT device usages internet connection for transmitting information to view and enables interactive speech recognition. This model enhances to the end-user demands as the capability of product bond extended.

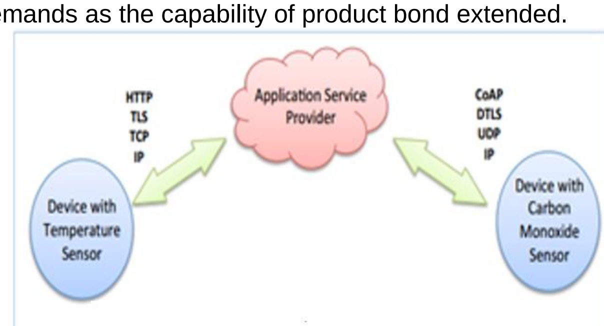

Fig.4. *Device-to-cloud communication*

Device-to-Gateway Model

The connection is being established between the devices to application layer and it helps to have transaction between the devices to cloud services. The cloud services cannot access directly so same medium is required to access it such as Fitbit, it needs a medium to send or fetch the data from cloud, so it uses a smartphone for connecting cloud service [21].

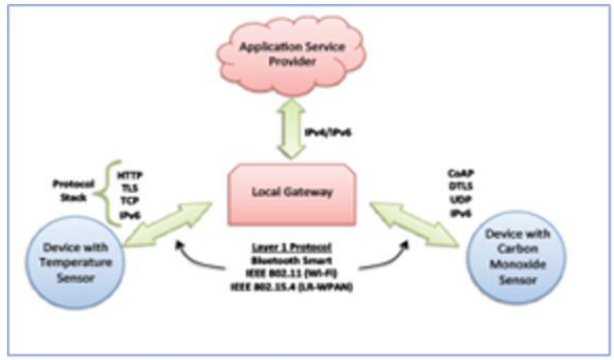

Fig.5. *Device-to-gateway communication*

Back end Data Sharing Model

The combination of data and other objects from cloud is exported and analysed by the users in this communication architecture.

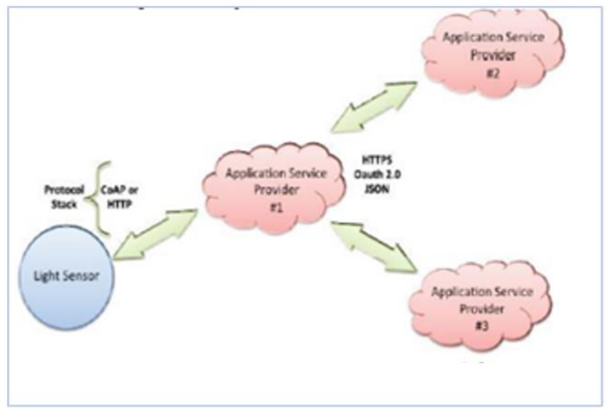

Fig.6. *Back end data sharing model*

The sensor data can be uploaded by the third party and this model is the leeway of single device-to-device communication.

4. CONCLUSION AND FUTURE WORK

Internet of Things is to secure the information while sharing with small business face problem such as storing a vast amount of data, etc. CONTIKI OS works under IoT platform and interface is based on JAVA for authenticity. The communication models supports CONTIKI OS for transmitting the data with an thwarting channel form one to another. In future author can focuses on the IoT architecture with middleware layer security and Risk architecture in IoT. These architectures helps to prevent from various attacks while communicating information from one node to another.

REFERENCE

[1] Shen, guicheng and bing wu liu , "The vision, technologies, application and security issues of IoT" , E- Business and E-Government, International conference on IEEE 2011.

[2] L. atzori, A. lera, G. morabito , "Internet of Things: A survey of computer networks" , No. 54(15) (2010) 2787-2805.

[3] Cerf, Vint, and Maureen ohlhausen , “Internet of Things” , FTC IoT workshop, Washington DC 19 nov 2013.

[4] IoT overview with the link: https://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151014_0.pdf.

[5] Scott brave, Hiroshi ishii and Andrew Dahley , “Tangible interface for remote collaboration and communication” , MIT media lab with Tangible media group.

[6] Ala-al-fuqaha, Mohsen guizani, Mehdi mohammadi, md. Aledhari , Moussa ayyash “Internet of Things: A survey on empowering technologies, protocols and application” , IEEE communication survey and tutorial, vol-17, no. 4, fourth quarter 2015.

[7] Krushang sonar, Hardik Upadhyay , “A survey of security privacy issues of internet of things”.

[8] C. buckl, S. sommer, A, scholz, A. kvoll, A. kemper, J. heveret al , “Service to the field: An approach for resource constrained sensor network” ,workshop of advance information network and application, presented in IEEE International conference on 2009, pp 968-975.

[9] M U Farooq, md. Waseem, Sadia mazhar, Anfum khairi , Kamal , “A review of IoT” , International journal of computer application (0975-8887) vol. 11, no.-3, 1, march,2015.

[10] Tschofenig, H., et. al., “Internet Architecture Board: Architectural Considerations in Smart Object Networking” ,Tech. no. RFC 7452., Mar. 2015.

[11] Tuhin Borgohain, Uday Kumar and Sugata Sanyal. , “Survey of Security and Privacy Issues of Internet of Things”.

[12] Presentation from bluetooth.com for the definition and detailed knowledge of it.

[13] Z-wave.com presentation for the definition and detailed knowledge of it.

[14] zigbee.com presentation for the definition and detailed knowledge of it.

[15] Duffy Marsan, Carolyn , “IAB Releases Guidelines for Internet-of-Things Developers” , IETF Journal, 11.1, July 2015, Web. Link: https://www.internetsociety. org/sites/default/files/Journal_11.1.pdf

[16] “Samsung Privacy Policy—Smart TV Supplement.” , Samsung corp.web , 29Sept2015, web link: www.samsung. com/sg/info/privacy/smart tv.

[17] “Internet of Things Global Standards Initiative" , ITU, 26 June 2015.

[18] “International Telecommunication Union: Overview of the Internet of things” , ITU-T Y.2060, June 2012.

[19] Pew ,“Report: An In-depth Look at Expert Responses", Research Centre of Internet, Science & Tech in 14 May 2014 and retrieved 26 June 2015.

[20] Mathilde Durvy, Julien Abeillé, Patrick Wetterwald, Colin O’Flynn, Blake Leverett, Eric Gnoske, Michael Vidales, Geoff Mulligan, Nicolas Tsiftes, Niclas Finne, and Adam Dunkels , “Making Sensor Networks IPv6 Ready” , ACM 6th Proceedings conference on Embedded network sensor systems, USA, 2008, pp. 421–422.

[21] A. Dunkels, O. Schmidt, T. Voigt, A. Muneeb , “Protothreads: Simplifying event-driven programming of memory constrained embedded systems” , ACM 4th Proceedings Conference on Embedded Networked Sensor Systems, Colorado, USA, 2006. Web link: http://dunkels.com/adam/dunkels06 protothreads.ppt. [Accessed:02-Mar-2014].

[22] A. Dunkels ,“Contiki: Bringing IP to Sensor Networks” , ERCIM News, no. 76, pp. 59–60, Jan- 2009. Web link: http://ercim-news.ercim.eu/images/stories/EN76/EN76-web.pdf.

[23] A. Dunkels, “Contiki operating system” , webpage link: http://www.sics.se/~adam/ Contiki/accessed in January 22, 2006.

[24] A. Dunkels, B. Grönvall, and T. Voigt., “Contiki: A lightweight and exible operating system for tiny networked sensors” , 1st Proceedings of IEEE in Embedded Networked Sensors Workshop, Tampa, Florida, USA, Nov. 2004.

[25] A. Dunkels, O. Schmidt, and T. Voigt , “Real world wireless sensor networks: Using Protothreads for Sensor Node Programming” , Proceedings Workshop of REALWSN, Stockholm, Sweden, June 2005.

[26] The “Contiki operating system 2.x online documentation” was given by A. Dunkels with a web page link: http://contiki.sourceforge.net/html, accessed January 22, 2006.

[27] A. Dunkels , “Full TCP/IP for 8 Bit Architectures” ,1st Proceedings of ACM/Usenix, Applications and Services (MobiSys), International Conference on Mobile Systems, San Francisco, May 2003.

[28] Timothy Malche, Priti Maheshwary , “Review: Harnessing the Internet of Things (IoT)” ,August 2015, International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 5, Issue 8, pp.320-323.

[29] The web link: http://www.tinyos.net/ for the detailed knowledge of Tiny OS, [Accessed: 13- May- 2016].

[30] The “Real-Time Kernels: µC/OS-II and µC/OS-III” with a detailed knowledge in the web link: https://www.micrium.com/rtos/kernels/, [Accessed: 13-May- 2016].

[31] The detailed knowledge of RIOT in the web link: https://www.riot-os.org/, [Accessed: 13- May- 2016].

[32] The detailed knowledge of Brillo in the web link: https://developers.google.com/brillo/, [Accessed: 13-May-2016]

[33] A.M. Vilamovska, E. Hattziandreu, R. Schindler, C. Van Oranje, H. De Vries, J. Krapelse, “ RFID Application in Healthcare – Scoping and Identifying Areas for RFID Deployment in Healthcare Delivery” , RAND Europe, February 2009.

[34] E. Welbourne, L. Battle, G. Cole, K. Gould, K. Rector, S. Raymer, M. Balazinska, G. Borriello , “Building the internet of things using RFID: the RFID ecosystem experience” , IEEE Internet Computing 13, 2009, 48–55.

[35] The web link: http://www.sensei-project.eu/ for “SENSEI FP7 Project” with Scenario of Portfolio, User and Context Requirements, Deliverable.