The Korea Academic Society of Digital Business Administration (KASDBA)
(사)한국디지털경영학회

ICIDB-2016 Conference Program

International Conference on Information and Communication Technology and Digital Convergence Business (ICIDB-2016)

Chung-Ang University, Seoul, South Korea, Dec 16-17, 2016

■ **Date:** Friday and Saturday, December 16-17, 2016 10:00am – 5:30 pm

■ **Place:** International Conference Hall (4th Floor), Graduate School Building, Chung-Ang University, Dongjak-gu, Seoul, South Korea

■ **Organizer:**

The Korea Academic Society of Digital Business Administration (KASDBA) (사)한국디지털경영학회

■ **Co-organizer:**

Korea Institute of Digital Convergence (KIDICO)

Social Science Research Institute, Yeungnam University

■ **Sponsors:**

National Information Society Agency

RIOT-OS: Operating System for Future IoTs

Illa Ul Rasool, Yousaf Bin Zikria, Arslan Musaddiq, Farhan Amin, Sung Won Kim*

Department of Information and Communication Engineering, Yeungnam University, South Korea

illaulrasool, yousafbinzikria, arslan@ynu.ac.kr,farhanamin10@hotmail.com, swon@yu.ac.kr

Abstract

Internet of Things is a technology for future. It is to provide a bridge between real world and electronic devices-realism. However, advantages of IoTs also accompany with constraints. These constraints span across low memory footprint, low-power CPU, etc. Therefore in IoT mitigating these constraints is a major issue. One solution to this issue is an efficient Operating System for IoT. Significantly, RIOT-OS for IOTs is of high importance for future development. This paper represents the RIOT-OS study with respect to more recent network technology, Named Data Networking. The study mainly considers the resource constraints of IoT devices and RIOT-OS features-implementation to mitigate them. Further, a combined architecture of NDN-RIOT with smart tick less timer scheduling mechanism to increase energy efficiency of devices is discussed. Other relevant features of RIOT supporting existing transport protocols are also studied.

Keywords—RIOT, NDN, CS, FIB, PIT.

I. Introduction

The use of technology and devices has reformed the way humans work. It is the modern phase of technology where electronic devices can no more afford to stay isolated and disconnected from the world. This motivation is driving the devices or things towards interconnectivity forming a network-of-things or Internet-of-things (IoT) [1]. IoT is a network of things which ranges between real time devices such as

Corresponding author*

Keywords- IoT, RIOT, CS,PIT,FIB.

wireless sensors, medical devices, security cameras, and even large infrastructures and vehicles. The significance of IoT powered devices is they can be accessed remotely from anywhere, as well as command other devices to perform necessary tasks. Although these devices increases the physical world awareness among devices are constrained in (i) available memory, (ii) processing power, (iii) communication, (iv) battery life time [2]. With all constrains considered, these devices are nevertheless required to adapt to the physical world and perform necessary real-time processing.

Acknowledging all the requirements of IoT devices and the related constrains, an IoT-Operating System (OS) must be capable enough to withstand such incompatible and varying hardware properties. RIOT OS is a solution to IoT devices spanning across low power to energy efficient microcontroller units. RIOT OS exhibits full support for IOT devices in terms of Minimum RAM, ROM requirement [4]. It also provides developer friendly environment for C and C++ programmers. RIOT-OS micro-kernel supports the core mechanisms such as multi-threading, priority-based scheduling interrupt handling, and inter-process communication. It is accurate synchronized with hardware to provide real-time execution of applications. For a summarized information, Table I provides a brief overview of RIOT-OS features. With all the features and requirements for IoT devices in check, this paper focuses on RIOT OS in terms of Named Data Networking (NDN) scheme [3]. Also referred as Content Centric Network (CCN), NDN is a Data-name based networking scheme. The motivation behind NDN is to achieve synchronism between internet architecture and its usage. It means that not all Internet services are sufficiently supported by the current built in architecture. The idea of NDN resides within the ambitious Information Centric Network (ICN) project which follows the content/information/ data centric approach [3]. The overall goal of data/content/information oriented network is to redefine the current host-centric Internet architecture (TCP/IP) into information-centric. It is clear enough that IoTs are expected to be implemented extensively in future. Also the

corresponding OS and wireless network are going to play an important role in its development and application process. NDN-RIOT for IoT devices itself define a total overhaul of present wireless technology into more convenient and less complex one. The design goal of NDN-RIOT is (i) to support the basic NDN forwarding mechanism, (ii) to support the current protocol regulations, (iii) NDN-RIOT OS must suit memory constrains of IoTs, (iv) full support for CPU that runs at a clock speed less than 100 MHz [4]. The contribution of our work introduces the RIOT OS for IoTs, presents the architecture for RIOT OS. Moreover analyzes the NDN-RIOT architecture-mechanism and features with an NDN application. This paper discusses in Section 2 the features of RIOT-OS for IoTs. Section 3 describes basic mechanism of NDN. Section 4 features NDN-RIOT architecture and example run of NDN in RIOT. Finally, Section 5 concludes the paper.

II. RIOT: Architecture

Earlier, the OS for wireless sensors and for Internet hosts differed with respect to available memory, energy efficiency, modularity of kernel, and API access. RIOT OS goal is to fulfill the varying requirements of IoTs that require overall OS reliability and availability of corresponding C and C++ libraries for developers. RIOT offers the real multi-threading processing inherited from FireKernel [4].Thus, FireKernel contributes to the real-time and modularity requirements of IoTs. It features zero-latency interrupt handlers, and minimum context-switching times along with threading priorities.

TABLE I. RIOT FEATURES [3]

FEATURES	RIOT OS
Minimum RAM	✓
Minimum ROM	✓
C Support	✓
C++ Support	✓
Multi-Threading	✓
MCU without Memory Management Unit	✓
Modularity	✓
Real-Time	✓

Moreover, every microcontroller unit has a particular scheduler which wakes up the system at certain time instants. Schedulers refer to the timers for such waking purpose. This process is called the timer tick. However, there can be certain instants when devices are not required to perform tasks and they are triggered by scheduler to wake up. To avoid this, RIOT introduces a tickless scheduler which allows the device to switch to idle state and enter deep sleep mode thus increasing the energy efficiency [5]. Adding to this, the devices can be interrupted in deep-sleep mode by an external interrupts or kernels only. Further, efficient energy output of an OS is dependent on the degree of complexity of kernel processes. Further, kernel functions depend upon the duration and occurrence of context switching. RIOT OS mainly performs the thread switching by an interrupt. Therefore, it is important to reduce the amount of thread switch-time-under-process. For that instant, RIOT introduces minimized scheduler, which after finishing the interrupt service routine does not requires saving old thread-context. This process significantly reduces task-switching processing time.

III. Named Data Network

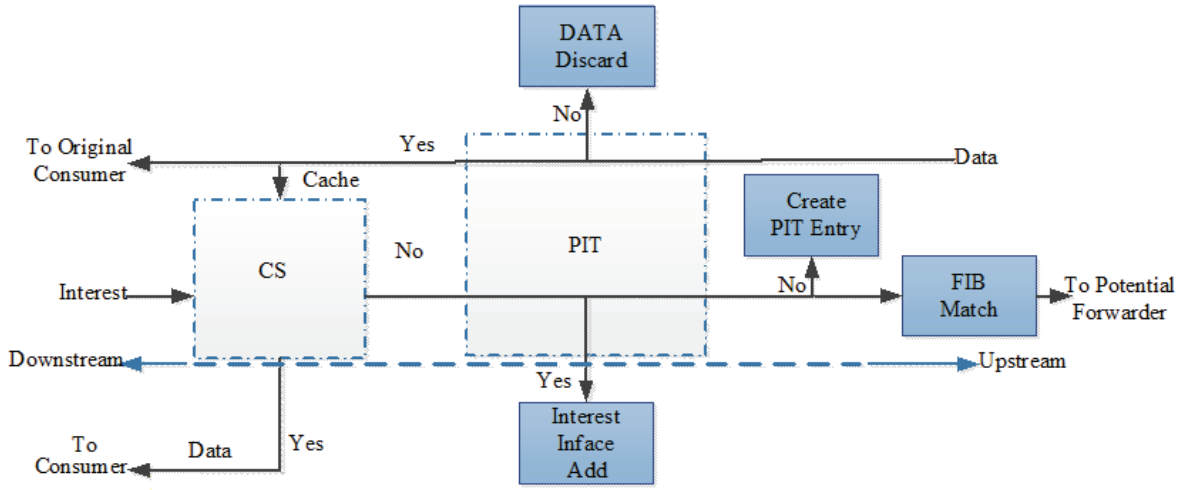

Fig.1. Named Data Network forwarding scheme.

The NDN communication follows a specified pattern or path to which the data plays the catalyst role rather than concentrating on host-client identities (IP-based). In NDN the content or data is defined by a specific name which regulates its recognition and retrieval by the

destination application. The NDN architecture constitutes of a consumer and producer. The consumer sends a data-request in the form of an Interest packet. Interest packet has a data-name as its prefix. The data packet is forwarded in the network following the Interest name and Interest Forwarding Strategy. The forwarding strategy is particularly based on the Forwarding Information Base (FIB). Every forwarding node has a FIB to make the calculative decisions about the potential destinations from where data can be retrieved. The data-destinations are actually the nodes that provide the requesting consumers with requested data. The node that satisfies the consumer interest is referred as Provider. Meanwhile, during the forwarding process of Interest packet each node in the routing path keeps the track of the interface from which the interest arrived in its Pending Interest Table (PIT). Also, each forwarder maintains a temporary data cache called Content Store (CS). CS is used to cache the data earlier sent to satisfy a specific Interest. Furthermore, an Interest can find its requested data either at the real Provider or in the CS of any forwarder. Finally when the Interest is satisfied, the data follows the same routing path as Interest packet earlier recorded in the PIT of every forwarder. Further, the data packet constitutes of a cryptographic signature that allows the consumer to authenticate the content. The main emphasis is not on the provider or node but the content or data produced by the provider. Fig.1. represents the basic NDN networking scheme and forwarding strategy.

IV. NDN-RIOT

a. Aim

Aim of NDN-RIOT OS is to fully support the named packet forwarding procedure of NDN, as well as it must obey the present protocol regulations. NDN is a futuristic technology and most probably will be implemented in IoTs.

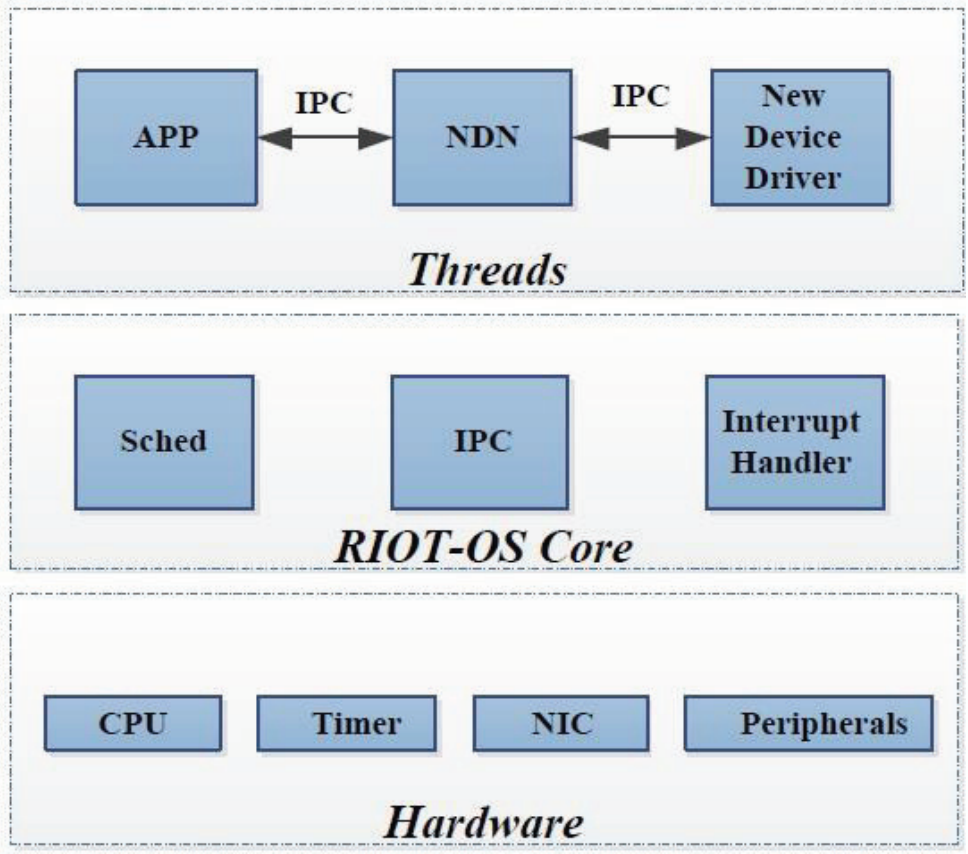

Fig.2.NDN-architecture and inter-process communication between threads.

NDN-RIOT supports such devices with 10s of Kb of RAM (executional data store), 100s of Kb of flash memory, and low-power Central Processing Unit running at a clock speed of less than 100 MHz. With such memory constraints the NDN-RIOT OS must execute and support all core mechanisms and functionalities of NDN such as FIB, PIT and CS.

b. Architecture

RIOT OS implements micro-kernel architecture as kernel threads. Fig.2, shows these threads represent three respective network protocols; IPv6, UDP. Inter-Process communication (IPC) is responsible for the communication between kernel threads. NDN-RIOT constitutes of threads; Application, NDN, Network device driver [4]. When an application requires to send a particular data, it forwards the packet to NDN layer via IPC. Further the packet is processed and transferred to network device driver for transmission. In case of packet reception, it follows the reverse path till packet reaches application layer. The above process occurs in the

presence of an interrupt handler, and other peripherals. RIOT OS handles the memory constraints in NDN- devices by storing the NDN packets in wire format throughout.

c. Packet Forwarding

The NDN communication follows a specified pattern or path to which the data plays the catalyst role rather than concentrating on host-client identities (IP-based). In NDN the content or data is defined by a specific name which regulates its recognition and retrieval by the destination application. In NDN, every forwarding node has a FIB to make the calculative decisions about the potential destinations from where data can be retrieved. The data-destinations are actually the nodes that provide the requesting consumers with requested data. The node that satisfies the consumer interest is referred as Provider. Meanwhile, during the forwarding process of Interest packet each node in the routing path keeps the track of the interface from which the interest arrived in its Pending Interest Table (PIT). Furthermore, each forwarder maintains a temporary data cache called Content Store (CS). CS is used to cache the data earlier sent to satisfy a specific Interest. Therefore, an Interest can find its requested data either at the real Provider or in the CS of any forwarder. Finally when the Interest is satisfied, the data follows the same routing path as Interest packet earlier recorded in the PIT of every forwarder. Further, the data packet constitutes of a cryptographic signature that allows the consumer authenticate the content. The main emphasis is not on the provider or node but the content or data produced by the provider. Fig.1. represents the basic NDN networking scheme and forwarding strategy. In basic NDN-forwarding strategy, an interest packet when received by any neighboring node performs the following process. Firstly, it checks the availability of data in its own CS following a lookup process. In case the data is found, it is dissipated back to the consumer through same interface the interest arrived earlier. However in case of no data found, node performs the PIT lookup. During PIT lookup process if a corresponding entry is found for the same data, it discards the interest and saves the incoming interface. Nevertheless in case of non-PIT matchup, a PIT entry is created for the interest and is further forwarded to the FIB.

Consequently FIB preforms calculative based decision to search for a potential data-provider. Secondly when interest finds a data match at a certain provider, a data packet under same interest-name-prefix match is transverse back. The data packet again follows the same interface/s on the way back to consumer. Moreover, on the way back each forwarder will authenticate the data by checking the corresponding PIT entry earlier stored. Further when data reaches the original consumer, it runs a signature validation test to confirm/authenticate the data received.

Example Run

We run a simulatory example for NDN-RIOT scenario, where we create five nodes to simulate basic NDN working in RIOT. Fig.3, shows how tap interface feature of Linux allows the ndn-ping application to see a raw network traffic. We assign tap 0 to consumer and tap 3 to producer nodes, respectively. Fig.4, shows how consumer expresses an interest in a NDN named format. Fig.5, shows the packet acknowledgement by producer and replies back with data after a named prefix match. On the reception of data by the consumer, it checks the data authentication after a signature validation process. In this manner RIOT is able to express the core functionality of NDN network by basing communication on data name rather host-centric (IP-based).

Fig.3. Node creation for scenario.

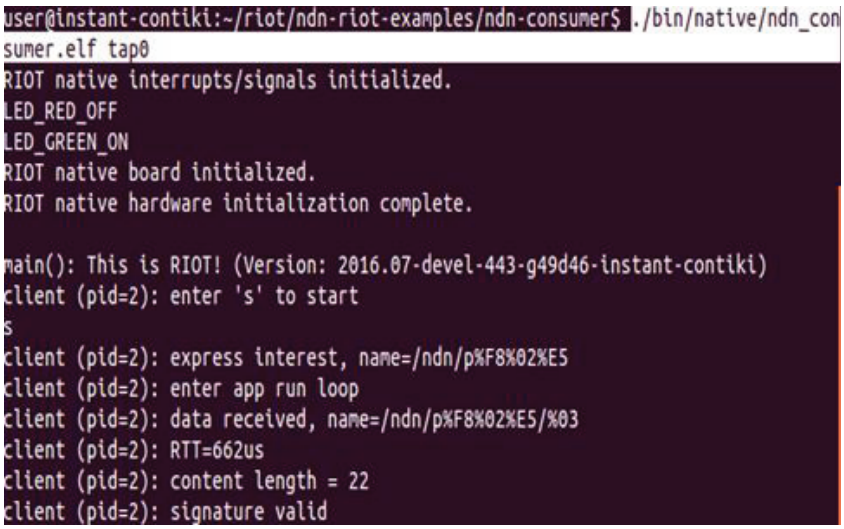

Fig.4. Consumer interest/data expressing/receiving.

Fig.5. Producer acknowledging Interest and sending data packet.

CONCLUSION

The basic architecture of RIOT-OS for IoT devices was briefly discussed. RIOT OS provides memory, and energy efficient features as well as modularity and real-time support for IoT devices. RIOT is also programmable and provides developer with C and C++ libraries. Further RIOT OS is studied in accordance with NDN, which is a futuristic wireless network technology. NDN-RIOT provides an overhaul of concept of how “Internet of Things” can work without any memory and

energy constraints in future. Further, the RIOT is compatible with the existing protocols (IPv6, RPL) only contributes for future convenience. The architecture for FIB, CS, PIT is a basic one implemented in RIOT due to memory constraints. However, comparatively NDN works efficiently with respect to RIOT and future work can include implementing full data names.

Acknowledgment

This research was supported in part by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2016-R2718-16-0035) supervised by the IITP (National IT Industry Promotion Agency) and in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2015R1D1A1A01058751).

References

Research papers

[1] Atzori L, Iera A, Morabito G, The internet of things: A survey, Computer networks, 2010 Oct 28;54(15):2787-805.

[2] A. Sehgal, V. Perelman, S. Kuryla, and J. Schonwalder. Management of resource constrained devices in the internet of things, Communications Magazine, IEEE, vol. 50, no. 12, pp. 144–149, 2012.

[3] Shang W, Afanasyev A, Zhang L. The Design and Implementation of the NDN Protocol Stack for RIOT-OS, NDN,Technical Report, NDN-0043, july 16, 2016.

[4] Baccelli E, Hahm O, Gunes M, Wahlisch M, Schmidt TC. RIOT OS: Towards an OS for the Internet of Things. InComputer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on 2013 Apr 14 (pp. 79-80). IEEE.

[5] Roussel K, Song YQ, Zendra O. RIOT OS Paves the Way for Implementation of High-Performance MAC Protocols. arXiv preprint arXiv:1504.03875. 2015 Apr 15.