DNSNA: DNS Name Autoconfiguration for Internet of Things Devices

Sejun Lee*, Jaehoon (Paul) Jeong† and Jung-Soo Park‡
* Department of Computer Science & Engineering, Sungkyunkwan University, Republic of Korea
† Department of Interaction Science, Sungkyunkwan University, Republic of Korea
‡ Electronics and Telecommunications Research Institute, Republic of Korea
Email: {sejunlee,pauljeong}@skku.edu, pjs@etri.re.kr

***Abstract*—This paper proposes a DNS Name Autoconfiguration (called DNSNA) for not only the global DNS names, but also the local DNS names of Internet of Things (IoT) devices. Since there exist so many devices in the IoT environments, it is inefficient to manually configure the Domain Name System (DNS) names of such IoT devices. By this scheme, the DNS names of IoT devices can be autoconfigured with the device's category and model in IPv6-based IoT environments. This DNS name lets user easily identify each IoT device for monitoring and remote-controlling in IoT environments. In the procedure to generate and register an IoT device's DNS name, the standard protocols of Internet Engineering Task Force (IETF) are used. Since the proposed scheme resolves an IoT device's DNS name into an IPv6 address in unicast through an authoritative DNS server, it generates less traffic than Multicast DNS (mDNS), which is a legacy DNS application for the DNS name service in IoT environments. Thus, the proposed scheme is more appropriate in global IoT networks than mDNS. This paper explains the design of the proposed scheme and its service scenario, such as smart road and smart home. The results of the simulation prove that our proposal outperforms the legacy scheme in terms of energy consumption.**

***Index Terms*—Internet of things, IPv6, DNS, autoconfiguration, device discovery, neighbor discovery**

I. Introduction

As one of the most spotlighted research areas these days, the Internet of Things (IoT) aims to provide users with various services through many devices. IoT-enabled devices can be remotely controlled and monitored across existing network [1]–[3]. Along with this trend, even in a small-scale home area in the future, there will be a lot of IoT devices with diverse performance capacity, ranging from high performance of computation and processing to only simple or limited communication ability. As IoT devices with high performance, in a home area, they are usually big appliances (e.g., smart TV, refrigerator, air conditioner, and washing machine). With WiFi module, high performance CPU, basic storage, and tailored OS, they are enabled to perform communication, sensing, computing, and actuating in their environments. Also, they let users be able to remotely control and monitor them in the Internet. For even low-capacity devices (e.g., light, meter, temperature controller, and sensors), they will still be useful by the enhanced and simplified management in their network or the Internet.

According to the annual Hype Cycle of Gartner [4] in 2014, IoT will be a promising technology over the next 10 years. Also, the number of IoT devices will increase to almost 26 billion units by 2020 [5]. Since there exist so many devices in IoT environments, it is inefficient to manually configure their domain names in the Domain Name System (DNS) which allows translation between domain names and IP addresses by DNS servers. Especially, this work is motivated by the observed trend that the DNS name of an IoT device can be autoconfigured with the device category and model without a user's intervention. The user can easily identify IoT devices by looking at the DNS name of each device for the purpose of monitoring and remote-controlling.

IPv6 was intended to replace IPv4 due to the limited address space [6]. Neighbor Discovery (ND) [7] is a protocol for IPv6. IPv6 host uses ND to find default gateway and determine the link-layer addresses of neighbors in the same subnet. ND uses Router Solicitation (RS) [8] and Router Advertisement (RA) [8] through Internet Control Message Protocol (ICMPv6) [9] for IPv6 stateless address autoconfiguration. When IPv6 host firstly joins a subnet, it sends an RS, which is an ICMPv6 message with type 133, to a router to ask an RA without waiting for the next RA interval. Each router periodically multicasts an RA, which is an ICMPv6 with type 134, with router information in the same subnet. The options in an RA [10] can include either Recursive DNS Server (RDNSS) for DNS name resolution or DNS Search List (DNSSL) for a list of domain suffixes for fully qualified domain name construction. With this IPv6, it is expected to have an easier way to manage numerous IoT devices [11]. Moreover, the automatic configuration of DNS name for each IoT device will significantly improve efficiency in the DNS name assignment. This DNS naming will let the users of IoT devices easily identify and manage them.

Bonjour [12] is the representative implementation of zero-configuration networking to support DNS naming and service discovery from Apple, Inc. In Bonjour, multicast DNS (mDNS) is used as a carrier protocol for DNS-based service discovery while no authoritative DNS server exists [13], [14]. However, since mDNS utilizes multicast for DNS name resolution in a local network, it is not suitable to apply it for global naming service in a multi-link subnet or a large-scale network because of heavy DNS traffic that is generated from multicast. In order to mitigate this kind of situation in terms of global DNS names, a scheme with light DNS traffic should be invented

due to an uncountable number of IoT devices in the future. Thus, for IPv6-based IoT environments, this paper proposes a scheme called DNS Name Autoconfiguration (DNSNA) that autoconfigures the DNS names of IoT devices and manages it for IoT devices' DNS naming. Based on DNSNA, IoT users in the Internet can easily distinguish the DNS names of IoT devices for monitoring and remote-controlling. Note that this paper is the enhanced version of our early paper [15]. The contributions of the paper are as follows:

- Global/local DNS name formats for IoT devices.
- A DNS naming framework for IoT devices.
- A DNS name generation scheme for an IoT device.
- A DNS name registration protocol for an IoT device.

The rest of this article is organized as follows. Section II summarizes related work. Section III describes our design of DNSNA. Section IV specifies our experiment method to evaluate our DNSNA. Section V describes the implementation of DNSNA in smart grid. In Section VI, we conclude this paper along with future work.

II. Related Work

Nowadays, IoT has been realized and popularly used for a variety of Internet services. The number of IoT devices in the networks has been steadily increasing. The naming system is an essential element to identify each IoT device in the network even though some of them have the same manufacturer and device type. Recently, due to this reason, the IoT research has paid a lot of attention to the naming and addresses for service discovery [16].

The oneM2M [16] develops technical specifications for a common M2M service layer. An M2M device identification scheme, which is based on object identifier (OID), was proposed by working group 2. This proposes a method for globally unique IDs to be used in oneM2M. The M2M node ID consists of a higher arc (i.e., M2M node indication ID) and a sequence of four arcs (i.e., manufacturer ID, model ID, serial ID, and expanded ID). The higher arc is assigned and managed by ITU-T and ISE/IEC. The manufacturer ID identifies the node manufacturer. The model ID identifies the node model. The serial ID identifies the node serial number. The expanded ID is an optional arc for the legacy device which operates under an M2M node.

EPCglobal [17] is a GS1 initiative that leads industry-driven standards of electronic product code (EPC) for radio frequency identification (RFID). EPC is an unique number to use 96-bit binary for identifiable entity. However, EPC should access the EPC information server to acquire information. In this case, it is inefficient to limit dynamic information renewal. In order to solve this problem, EPCglobal suggests an object naming service (ONS). ONS supports global retrieval service because it can be converted to a uniform resource locator (URL). Since the URL is mapped to an IP address, it is possible to get the object's information. However, ONS is currently limited to a product type for resolution.

In [18], the authors propose an architecture of name service system in Mobile Ad-hoc Network (MANET). This approach makes a unique domain name with user-id or device-id (i.e., the EUI-64 identifier of the network devices MAC address) and DNS suffix (i.e., adhoc). However, the disadvantage of this naming is the low readability to human users in the retrieval of the device information.

Bonjour [12] is Apple's implementation of zero-configuration networking to support DNS naming and service discovery. Bonjour can support the search and registration of apple devices in order to announce the services of an Apple device. mDNS [13] is used by every device as both a DNS resolver and DNS server. mDNS is used as a carrier protocol for DNS based service discovery and no authoritative DNS server is not used for name resolution. mDNS uses multicast for name resolution in a local network, however, it is not suitable to a multi-link network because of many traffic. mDNS is a privacy issue because it supports a naming without the security key. DNS based service discovery protocol (DNS-SD) [14] is used to perform service discovery by Bonjour software. DNS-SD can provide a list of service devices and port numbers for hosts.

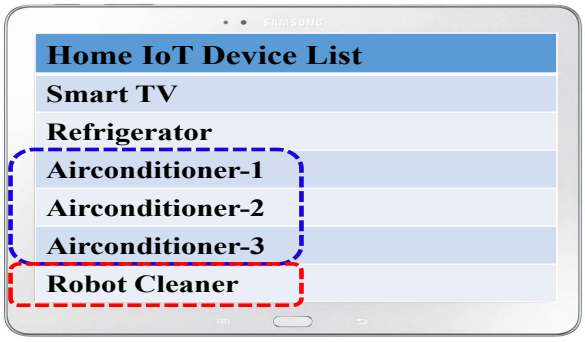
In this paper, we propose a DNS name autoconfiguration scheme for DNS name generation and resolution of IoT devices in an IoT network, such as home network and road network. Since our scheme works in unicast with a dedicated DNS server in the IoT network, it can provide an efficient DNS name services for IoT home devices. Also, since autoconfigured DNS names contain the device category and device model of the devices, home residents can easily identify devices with their DNS names.

III. DNS Name Autoconfiguration

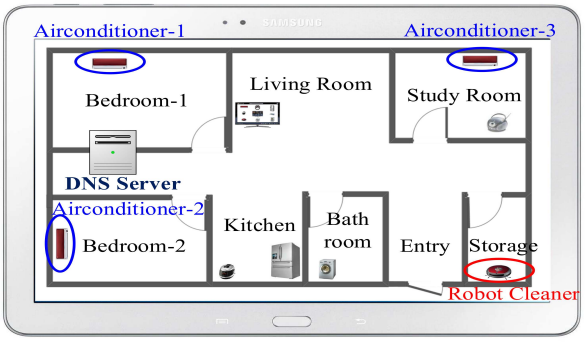
In this section, we introduce the architecture and scenario for DNS names of IoT devices. The DNS name autoconfiguration uses IPv6 ND to acquire DNS search list through either RA [10] or DHCPv6 [19]. Once a DNS search list is obtained, the target network device autonomously constructs its DNS name with the DNS search list and its device information.

A. Device Discovery

The device discovery in our paper displays the category (e.g., refrigerator and sensor), model (e.g., a manufacture's product model), and location (i.e., the place having an IoT device) of IoT devices for smart devices, such as smartphone and tablet PC. Fig. 1 illustrates the device discovery using Tablet PC in a home network. Fig. 1(a) shows the text list of IoT devices by tablet PC. In this case, it is hard to identify IoT devices and takes time for searching and identifying a device. For example, since there exist three air conditioners in the apartment, it is hard to identify them only with the list of IoT devices. Therefore, we consider the visual display of IoT devices. In this way, it is easy to identify IoT devices and takes instant time for searching and identifying a device, as shown in Fig. 1(b). For a robot cleaner, the graphic-based IoT device display in Fig. 1(b) can track its position in the apartment. On the other hand, the text-based IoT device display in Fig. 1(a) cannot do such a tracking.



(a) Text-based IoT Device List



(b) Graphic-based IoT Device Display

Fig. 1. Management of IoT Devices based on DNS Naming Service

B. DNS Name Format

A global DNS name for an IoT device has the following format, as shown in Fig. 2(a):

- **unique_id** is a unique identifier to guarantee the uniqueness of the DNS name in ASCII characters. The identifier may be a sequence number or alphanumeric with readability, such as product name.
- **device_model** is a device's model name in ASCII characters. It is a product model name provided by the manufacturer.
- **device_category** is a device's category name in ASCII characters, such as TV, smart car, tablet, light, and meter.
- **location** is a device's physical location, such as macro location (e.g., living room in an apartment) and micro location (e.g., the center of the living room). This field is optional (i.e., static device or dynamic device), so it can be excluded from the DNS name format.
- **domain_name** is a DNS domain name (e.g., skku.edu) that is encoded according to the specification of "Representation and use of domain name" [20].

unique_id.device_model.device_category.location.domain_name

(a) DNS Name Format using Device Information

unique_id.object_identifier.location.domain_name

(b) DNS Name Format using Object Identifier

Fig. 2. DNS Name Formats for DNSNA Protocol

We can include an OID of oneM2M [16] as part of the DNS name format for a global DNS name of an IoT device in the Internet. The OID is developed by ITU-T [21] and ISO/IEC [22] to assign a globally unique ID to an M2M node. A global DNS name with an OID for an IoT device has the following format, as shown in Fig. 2(b):

- **unique_id** is a unique identifier to guarantee the uniqueness of the DNS name in ASCII characters.
- **object_identifier** takes the form M2M node indication ID, manufacturer ID, model ID, and serial number ID [21], [22].
- **location** is a device's physical location. This field is optional, it can be excluded from the DNS name format.
- **domain_name** is a DNS domain name (e.g., skku.edu) that is encoded according to the specification.

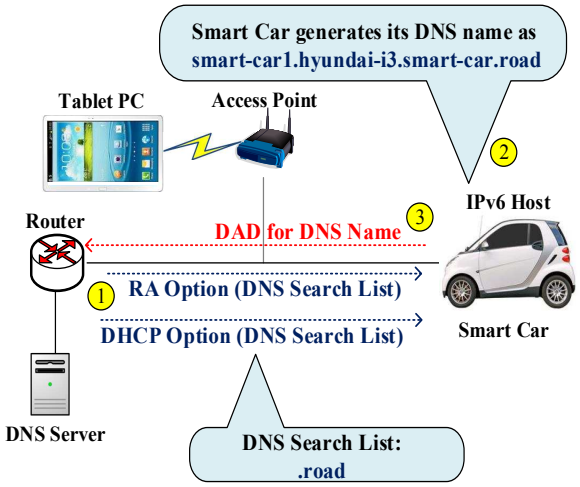


Fig. 3. IoT Device DNS Name Generation

C. DNS Name Generation

This section explains the procedure of DNS name generation, as shown in Fig. 3. When an IPv6 host such as smart car joins a subnet, it receives a DNSSL option through either RA from a router or DHCPv6 from a DHCPv6 server. The DNSSL is a list of DNS suffixes that are used for the construction of the fully qualified DNS names of IPv6 nodes. The IPv6 host checks the validity of the DNSSL option. If the option is valid, the IPv6 host performs the DNS name autoconfiguration for each DNS suffix domain name and a valid composition of a host name. As shown in Fig. 3, the host constructs a global DNS name format in Fig. 2(a), smart-car is unique ID for the product name, hyundai-i3 is device model about the product model, smart-car is device category, and road is domain name. The host performs Duplicate Address Detection (DAD) to test the uniqueness of the address corresponding to the global DNS name in the link where the host is located. Note that the global DNS name is converted into an IPv6 multicast address. If the host receives no response for the DAD, the global DNS name is regarded as unique in the link. It sets the corresponding

IPv6 tentative address as its IPv6 unicast address along with the global DNS name. Otherwise, when the host receives a response for the DAD, it should regenerate a new global DNS name with another identifier for the sequence number of the unique ID.

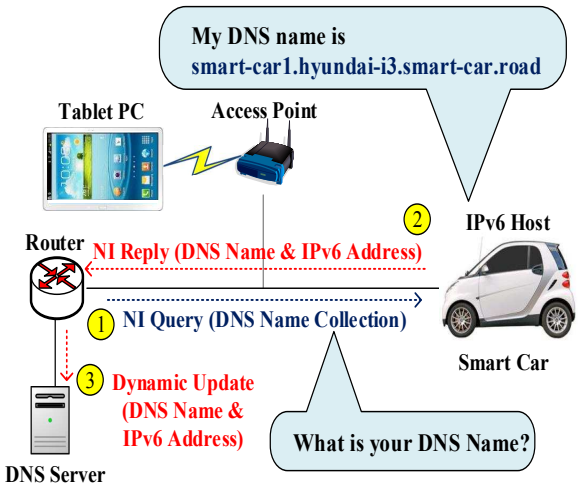


Fig. 4. IoT Device DNS Name Registration

D. DNS Name Registration

This section explains the procedure of DNS name registration, as shown in Fig. 4. Once the IPv6 host constructs its global DNS name. The IPv6 node (i.e., router or host) in the same subnet can collect the DNS name of IPv6 host (e.g., smart car) by using Node Information (NI) protocol [23]. The NI query is not defined yet for the collection of the DNS name of an IPv6 host. It should defined a new NI query or reply code (i.e., 3). The router sends an IPv6 host in order to collect the DNS name through an NI query in the same subnet. The query should be transmitted by the IPv6 node (i.e., router or host) to a link-local multicast address for this NI query. Assume that all the IPv6 hosts join multicast group for collection service of IPv6 host DNS name. The IPv6 host replies to the NI query. The IPv6 hosts need to send an NI reply to prevent message collision with a random interval between zero and Query Response Interval. As shown in Fig. 4, the collector is assumed as a router. The router checks the validity of the global DNS name of the host by sending a DNS query for the DNS name to an authoritative DNS server. If there is no corresponding IPv6 address for the queried DNS name, the router registers the DNS information (e.g., the global DNS name and the corresponding IPv6 address) of the IPv6 host into the authoritative DNS server for the domain name through DNS dynamic update [24]. The authoritative DNS server manages DNS zone file in order to translate between domain names and IPv6 addresses. Otherwise, a router notifies the IPv6 host of the duplication DNS name error. In this case, the host should regenerate its new DNS name.

E. DNS Name Retrieval

This section explains the procedure of DNS name retrieval. The authoritative DNS server stores the DNS information of

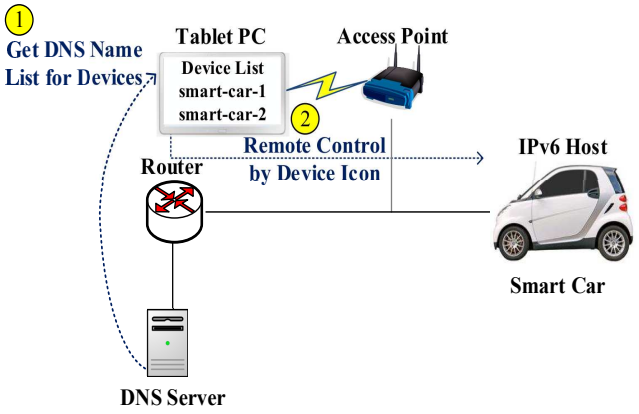


Fig. 5. IoT Device Management

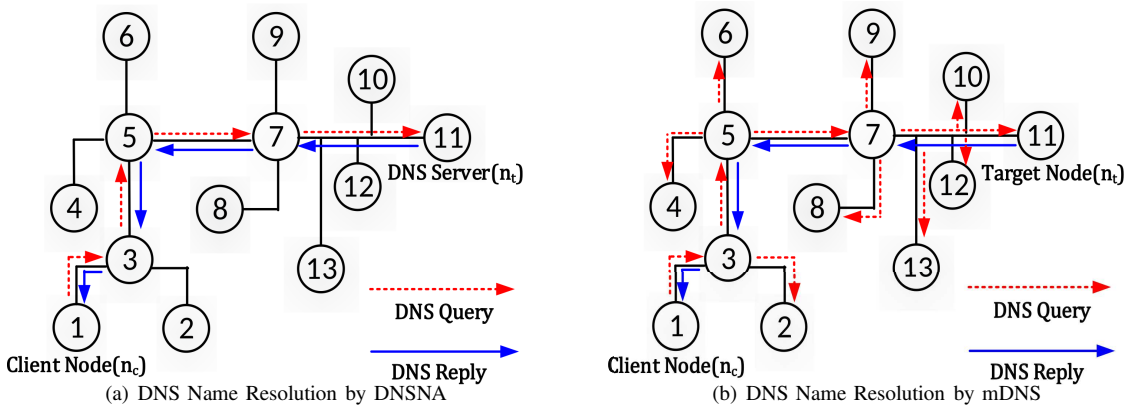
IoT devices through a zone file. The user gets the DNS name to manage the IoT devices. As shown in Fig. 5, the authoritative DNS server can transmit a DNS information of zone file into a user's smart device (e.g., smartphone and tablet PC). The user uses the smart device to display the DNS names of the IoT devices. These DNS names let IoT users (e.g., home residents and customers) in the Internet easily identify each device for monitoring and remote controlling in the target network.

F. DNS Name Management for Mobile IoT Devices

Some IoT devices can have mobility, such as smartphone, tablet, laptop computer, and robot cleaner. This mobility allows the IoT devices to move from a subnet to another subnet where subnets can have different domain suffixes, such as living room.home and garage.home. The DNS name change (or addition) due to the mobility should be considered. To deal with DNS name management in mobile environments, whenever an IoT device enters a new subnet and receives DNS suffix domain names, it generates its new DNS names and registers them into a designated DNS server, specified by RDNSS option. When the IoT device recognizes the movement to another subnet, it can delete its previous DNS name(s) from the DNS server having the DNS name(s), using DNS dynamic update. For at least one DNS name to remain in a DNS server for the location management in Mobile IPv6, the IoT device does not delete its default DNS name in its home network in Mobile IPv6.

IV. Performance Evaluation

In this section, we analyze performance of DNSNA, comparing with mDNS scheme for energy consumption. Fig. 6 shows the example topology of multi-link network. Our scheme uses unicasting for DNS name resolution service from client node to a designated node (i.e., DNS server or target node) in a multi-link network. However, mDNS has no authoritative DNS server, uses multicasting for DNS name resolution from client node to target node. Let $G = (V, E)$ be a target network graph where V is a vertex set (i.e., routers and hosts) and E is an edge set (i.e., links). The graph diameter (denoted as d_G) from



(a) DNS Name Resolution by DNSNA (b) DNS Name Resolution by mDNS

Fig. 6. Comparison between DNSNA and mDNS in terms of Traffic in Multi-link Network

client node (n_c) to the designated node (n_t) maximum hops is as follows:

$$d_G = d(n_c, n_t). \quad (1)$$

The graph diameter of d_G is the maximum hops between a pair of vertices in the graph. In DNSNA, the graph diameter of d_G is assumed to be m hops from client node to DNS server, as shown in Fig. 6(a). The worst-case in our scheme has the message number of 2*m due to unicasting. Thus, the overhead for DNS name resolution is O(m). On the other hand, in the worst case, mDNS may multicast DNS queries to the whole network with $|E|$ links. As shown in Fig. 6(b), mDNS for resolution generates the queries corresponding to the number of the total links ($|E|$) in the target network due to multicasting. mDNS generates m messages of the reply from target node to client node. Thus, the overhead for DNS name resolution is O($|E|$).

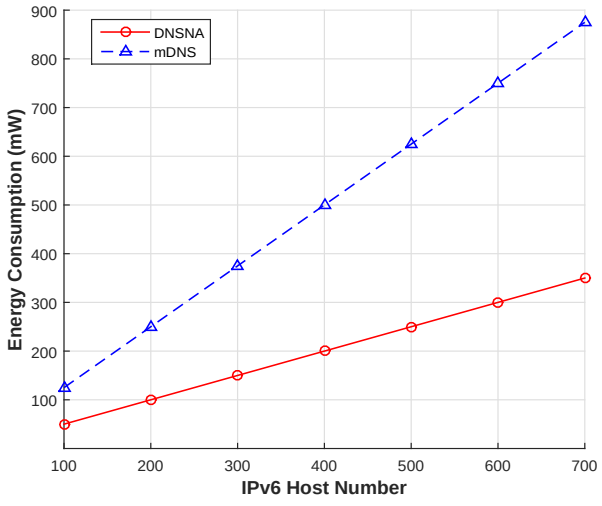


Fig. 7. Energy Comsumption for DNS Name Resolution

The evaluation setting is as follows:

- **Performance Metric:** We use energy consumption for the performance, which is proportional to the number of generated messages.

TABLE I
COMPARISON OF DNSNA AND MDNS

Approaches	DNSNA	mDNS
Packet Forwarding	Unicasting	Multicasting
Authoritative DNS Server	Yes	No (host itself is server)
Naming Scope	Global, Local	Local
Target Networks	Small, Large	Small
Target Devices	IoT Devices	Apple Products
The graph diameter for DNS Name Resolution	*O(m)*	*O(|E|)*
Resource Usage (CPU, Memory, Energy)	Small	Big

- **Baseline:** mDNS is used as a baseline, which is a popular legacy scheme.
- **Parameter:** On the performance, we investigate the number of nodes from 100 to 700.

The overhead of DNSNA and mDNS for DNS name resolution is evaluated using Matlab. We assume that energy consumption is 1 milliwatt per message and that both DNSNA and mDNS have the same path links from client node to a designated node (i.e., DNS server or target node). Fig. 7 shows the energy consumption as the number of IPv6 hosts increases. Both DNSNA and mDNS consume more energy as the number of IPv6 hosts increases. As the number of IPv6 hosts as IoT devices increases from 100 to 700, DNSNA has less energy consumption than mDNS due to the unicast scheme for DNS name resolution service. The two approaches of DNSNA and mDNS are compared for global DNS naming in Table I. As the table shows, since DNSNA does not use multicasting for packet forwarding, it can save more energy than mDNS. DNSNA can support both global and local DNS naming efficiently. However, mDNS can support local DNS naming efficiently.

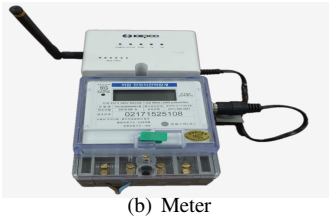
Therefore, it is shown that DNSNA is a promising global DNS naming system for IoT devices in a large-scale IoT network.

V. EXPERIMENT IN TESTBED

We have implemented DNSNA for the verification in a smart grid environment. We have developed our DNSNA protocol for the existing gateway and meter for a DNSNA testbed, as shown in Fig. 8.



(a) Gateway



(b) Meter

Fig. 8. Demonstration Devices

TABLE II
SPECIFICATION OF GATEWAY AND METER

Parameter	Gateway	Meter
OS	Linux	FreeRTOS
CPU	ARM Cortex-A5	ARM Cortex-M4F
RAM	DDR2 256M-byte	64K-byte embedded SRAM
FlashMemory	256M-byte	256/512Kbytes

Table. II shows the specification of gateway and meter. A gateway is equipped with ARM cortex-a5 core, DDR2 256M-byte, and 256M-byte flash memory. A meter is equipped with ARM cortex-M4F core, 64K-byte embedded SRAM, 256/512 Kbytes embedded flash memory. These devices are used in a real smart grid in Korea Electric Power Corporation (KEPCO) for a pilot service of remote meter-reading.

```
Interface wisun0
{
        AdvSendAdvert on;
        MinRtrAdvInterval 600;
        MaxRtrAdvInterval 1800;

        Prefix 2001:2c8:ee:100::/64
        {
                AdvOnLink on;
                AdvAutonomous on;
                AdvRouterAddr off;
        };
                                        RDNSS 2001:2c8::1
                                        {
                                         AdvRDNSSLifetime 3600;
                                        };
                                        DNSSL secter1.grid
                                        {
                                         AdvDNSSLifetime 3600;
                                        };
                                };
```

Fig. 9. Configuration of Router Advertisement Daemon

The Router Advertisement Daemon (radvd) [25] is an open-source software which is used as an IPv6 router. The configuration of DNSNA has the list of interface specific options, as shown in Fig. 9. radvd also supports the RDNSS [10] and DNSSL [10] options. In our demonstration, the domain name of the smart grid, which is delivered by the DNSSL option, is secter1.grid.

```
==========================================================
* Recv ND RA     Len=80
==========================================================
nd_ra option type:  3...len 4
nd_ra option type: 31...len 3
nd_ra dnssl data : 00 07 73 65 63 74 65 72 31 04 67 72 69 64 00 00
Write DNSSL Name : secter1.grid to ./dnssl_name.txt
Make Autoconf DNS Name:gtype-meter-cc14.jubix-wagmcc.powermeter.secter1.grid
```

Fig. 10. Meter DNS Name Generation

Fig. 10 shows the DNS name generation of a meter. The meter receives an RA through radvd. After this DNS name generation, to test the uniqueness of the generates DNS name, DAD is performed for the IPv6 multicast address corresponding to the tested DNS name.

```
==========================================
* Send NI Query   Len=32
    To    ff02::1
==========================================
* Type       : 139,  Node Information Query
* Code       : 0,  Query Subject is IPv6 Address
* Checksum   : 0x0000
* Qtype      : 2,  Node Name
* Flags      : 0x0000
* Nonce      : 0x4139418843843a8d
* IPv6 Addr  : ff02::1
==========================================
```

Fig. 11. Node Information Query for DNS Name Collection

Fig. 11 shows NI query for DNS name collection. The gateway sends the meter an NI query to collect the DNS name through NI protocol in the same subnet, after that the meter receives an NI query from the gateway.

```
==========================================================
* Recv NI Reply   Len=76
    From    fe80::ba27:ebff:fe62:60b2
==========================================================
* Type         : 140,  Node Information Reply
* Code         : 0,  Reply is Success
* Checksum     : 0x0000
* Qtype        : 2,  Node Name
* Flags        : 0x0000
* Nonce        : 0x38b93769386b36c4
* TTL          : 0
* Node Name : gtype-meter-cc14.jubix-wagmcc.powermeter.secter1.grid
==========================================================
```

Fig. 12. Node Information Reply for DNS Name Collection

Fig. 12 shows NI reply for DNS name collection. The gateway receives an NI reply about DNS information from the meter, after that the meter sends the gateway an NI reply.

```
Found zone name : secter1.grid
The master is : ns.secter1.grid
Sending update to 115.145.178.190#53
Outgoing update query :
;; ->>HEADER<<- opcode : UPDATE, status : NOERROR, id : 32518
;; flags ;; ZONE : 1, PREREQ : 0, UPDATE : 1, ADDITIONAL : 1
;; UPDATE SECTION :
gtype-meter-cc14.jubix-wagmcc.powermeter.secter1.grid. 300 IN AAAA  fe80::ba27:ebff:fe34:8bec
```

Fig. 13. DNS Dynamic Update

Fig. 13 shows DNS dynamic update. The gateway checks the validity of a global DNS name of the meter through the DNS name query to an authoritative DNS server. If there is no corresponding IPv6 address for the queried DNS name, the gateway registers the DNS information (e.g., DNS name and IPv6 address) of the meter into the authoritative DNS server managing the DNS domain through DNS dynamic update.

```
> gtype-meter-cc14.jubix-wagmcc.powermeter.secter1.grid
Server : 115.145.178.190
Address : 115.145.178.190#53

gtype-meter-cc14.jubix-wagmcc.powermeter.secter1.grid has AAAA address  fe80::ba27:ebff:fe34:8bec
```

Fig. 14. DNS Name Lookup

Fig. 14 shows the result of DNS name lookup for the global DNS name registered into the authoritative DNS server. Client PC can search for the DNS names of the IoT devices from the authoritative DNS server. The authoritative DNS server manages the DNS zone file for the IoT devices in order to translate between the domain names and IPv6 addresses of the IoT devices.

VI. Conclusion

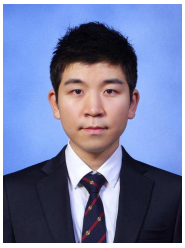
In this paper, we proposed our design of DNS Name Autoconfiguration (called DNSNA) for IoT environments. Our goal is to provide a reliable, efficient global DNS name autoconfiguration of IoT devices with the device information and without the intervention of human users. This DNS naming lets IoT users easily identify each device for monitoring and remote controlling in target networks, such as smart home and smart road networks. As future work, we will enhance DNSNA for IoT devices with privacy and security protection.

Acknowledgment

This research was supported by Basic Science Research Program (2014006438) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT &Future Planning (MSIP). This work was also partly supported by the ICT R&D program of MKE/KEIT (10041244, SmartTV2.0 Software Platform), MSIP/IITP (R0166-15-1041, Standard Development of Network Security based SDN), MSIP/IITP ICT/SW Creative Research Program R2215-15-1005, the SW Oriented College Support Program), and the "Leaders in Industry-university Cooperation" Project (2015-A-0002-010113), supported by the Ministry of Education. Note that Jaehoon (Paul) Jeong is the corresponding author.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities ," *Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.
[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things(iot): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep 2013.
[3] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Internet of Things Journal*, vol. 54, no. 15, pp. 2787–2805, Oct 2010.
[4] Z. Zhang, M. Cho, Z. Wu, and S. Shieh, "Identifying and Authenticating IoT Objects in a Natural Context," *IEEE Computer Society*, vol. 48, no. 8, pp. 81–83, Aug 2015.
[5] Gartner, Inc., "Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020," http://www.gartner.com/newsroom/id/2636073.
[6] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *IETF RFC 2460*, Dec. 1998.
[7] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," *IETF RFC 4861*, Sep. 2007.
[8] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," *IETF RFC 4862*, Sep. 2007.
[9] A. Conta, S. Deering, and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," *IETF RFC 4443*, Mar. 2006.
[10] J. Jeong, S. Park, L. Beloeil, and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration," *IETF RFC 6106*, Nov. 2010.
[11] A. Jara, L. Ladid, and A. Skarmeta, " The internet of everything through ipv6: An analysis of challenges, solutions and opportunities," *J.Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.*, vol. 4, no. 3, pp. 97–118, Sep 2013.
[12] Apple, "Bonjour," https://developer.apple.com/.
[13] S. Cheshire and M. Krochmal, "Multicast DNS," *IETF RFC 6762*, Feb. 2013.
[14] S. Cheshire and M.Krochmal, "DNS-Based Service Discovery," *IETF RFC 6763*, Feb. 2013.
[15] S. Lee and J. Jeong, "Design and Implementation of DNS Name Autoconfiguration for Internet of Things Devices," *Journal of KIISE*, Oct 2015.
[16] M2M, "Object Identifier based M2M Device Identification Scheme," http://www.onem2m.org.
[17] GS1, "EPCglobal," http://www.gs1.org/epcglobal.
[18] J. Jeong, J. Park, H. Kim, and K. Park, "Name Service in IPv6 Mobile Ad-hoc Network," *Lecture Notes in Computer Science (LNCS)*, vol. 2662, pp. 692–701, Aug 2003.
[19] R. Droms, "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," *IETF RFC 3646*, Dec. 2003.
[20] R. Droms, J. Bound, B. Volz, , C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," *IETF RFC 3315*, Jun. 2003.
[21] ITU-T Study Group 17, "General procedures and top arcs of the international object identifier tree," http://www.itu.int/itu-t/recommendations/rec.aspx?rec=X.660.
[22] ISO/IEC 9834-1, "General procedures and top arcs of the international object identifier tree," http://www.iso.org/iso/catalogue_detail.htm?csnumber=58055.
[23] M. Crawford and B. Haberman, "IPv6 Node Information Queries," *IETF RFC 4620*, Aug. 2006.
[24] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)," *IETF RFC 2136*, Apr. 1997.
[25] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," *IETF RFC 2462*, Dec. 1998.



Sejun Lee is a master student in the Department of Computer Science and Engineering at Sungkyunkwan University in Korea. His M.S. advisor is Professor Jaehoon (Paul) Jeong. He received the B.S. degree from the School of Information and Communication Engineering at Mokpo National Maritime University in Korea, in 2014. His research areas are Vehicular Ad-hoc Networks (VANET), Cyber-Physical Systems (CPS), and Internet of Things (IoT).



Jaehoon (Paul) Jeong is an assistant professor in the Department of Software at Sungkyunkwan University in Korea. He received his Ph.D. degree in the Department of Computer Science and Engineering at the University of Minnesota in 2009. He received the B.S. degree in the Department of Information Engineering at Sungkyunkwan University and the M.S. degree from the School of Computer Science and Engineering at Seoul National University in Korea, in 1999 and 2001, respectively. His research areas are IoT, CPS, vehicular networks, wireless sensor networks, software-defined networking, and network security.



Jungsoo Park works for Electronics and Telecomunications Research Institute (ETRI) as the Principle Researcher since 1994. He got his Ph.D. degree from the Department of Electronics Engineering at Kyungpook National University (KNU) in 2013. He got the B.S. degree and the MS degree from the Department of Electronics Engineering at Kyungpook National University (KNU) in 1992 and 1994, respectively. His research interests are network security, IoT, machine to machine (M2M), network functions virtualization (NFV), vehicular networks, wireless sensor networks, and mobile ad hoc networks.