사물인터넷을 위한 DNS 네이밍 서비스 기술동향

이근태, 정재훈
성균관대학교

요 약

본고에서는 최근 가장 주목 받고 있는 연구 분야 중 하나인 사물인터넷(Internet of Things, IoT)을 위한 Domain Name System(DNS) 네이밍 서비스 기술동향에 대해 소개한다. IoT 환경에서 IoT 디바이스를 관리하기 위해 DNS 네임을 일일이 수동적으로 설정하는 것은 비효율적이다. 따라서 본고에서는 IoT 환경에서 수많은 IoT 디바이스의 관리를 위한 IoT 디바이스의 DNS 네임 자동설정 및 네이밍 서비스 기술에 대해 소개하고 분석한다. 본고에서 소개하는 IoT 네이밍 서비스 기술은 Internet Engineering Task Force(IETF)에서 제정된 표준 프로토콜을 이용하므로 구현이 용이하고 성능적인 면에서도 우수하다. IoT 디바이스의 DNS 네임이 사용자들에게 가독성 있으면서도 기계적으로 파싱이 용이하면 IoT 디바이스 관리를 효과적으로 수행할 수 있다. 즉 설정된 IoT 디바이스 DNS 네임으로 인터넷 사용자가 스마트폰과 태블릿과 같은 모바일 스마트 디바이스로 IoT 디바이스를 원격에서 모니터링 및 원격제어 할 수 있다. 본고에서는 IoT 디바이스에 연관된 네이밍 서비스 기술들을 비교 분석하고, 본 저자가 제안하는 IoT 디바이스 DNS 네이밍 기술을 소개하고자 한다.

I. 서 론

사물인터넷(Internet of Things, IoT)은 최근 가장 주목 받고 있는 인터넷 관련 기술로 스마트 디바이스와의 연동을 통해 주변 센서로부터 수집된 정보를 가공 및 처리하여 다양한 서비스를 제공한다[1]. 미국의 시장조사기관인 가트너는 2014년도에 발표한 Hype Cycle에서 향후 5년에서 10년사이에 가장 유망한 미래의 기술로 IoT를 선정했고, 2018년도까지 80억개의 IoT 디바이스가 증가해서 총 160억개의 IoT 디바이스가 존재할 것이라고 발표했다[2].

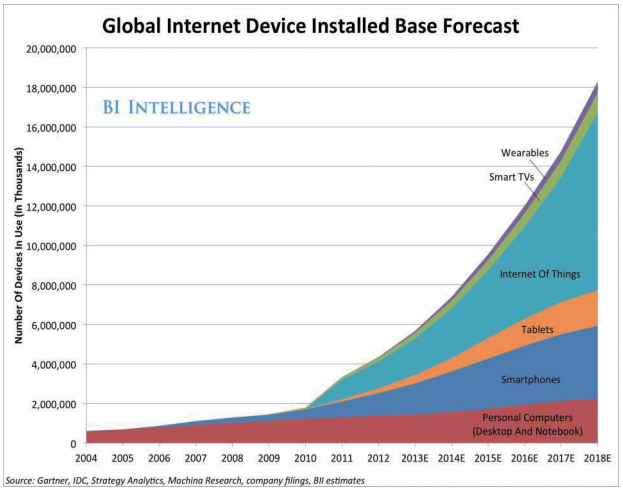


그림 1. 시간에 따른 IoT 디바이스 증가량[2]

IoT 디바이스가 인터넷에 연결되어서 서비스를 제공하고 관리되기 위해서는 Internet Protocol(IP) 주소가 있어야 한다. 현재 인터넷에서 사용되는 IP주소(예, 115.145.178.185)는 4바이트의 정수로 구성되며, 일련의 숫자와 점들로 표현된다. 4바이트의 주소체계를 이루는 현재 인터넷인 Internet Protocol Version 4(IPv4)는 인터넷 주소 고갈이라는 한계점이 있다. 그래서 이를 극복하기 위한 기술로 16 바이트의 주소체계를 가지는 Internet Protocol Version 6(IPv6)가 제정되었다[3]. 하지만 인터넷 상의 서버 머신에 해당하는 IP 주소를 직접 입력한다면 사용에 불편함이 많을 것이다. 따라서 이러한 IP주소를 사람에게 익숙한 레이블의 조합인 도메인 네임으로 연결하여 이 둘 간의 매핑을 제공하는 Domain Name Systme(DNS)이 제안되어 사용되고 있다. 즉 DNS는 사용자가 IP 주소를 쉽게 기억할 수 있는 문자열로 대응시키는 서비스로써 인터넷에 연결된 개체에 대한 IP 주소와 DNS 네임 간의 변환 서비스를 제공한다.

하지만 수많은 디바이스가 존재하는 IoT 환경에서 디바이스에 대한 DNS 네임을 일일이 수동적으로 설정하는 것은 매우 번거로운 일이다. 이를 해결하기 위한 방안으로 애플(Apple Inc.)에서 개발한 봉주르(Bonjour)는 무설정 서비스인 제로콘

프(Zeroconf)를 지향하는 소프트웨어인 multicast Domain Name System(mDNS)를 통해 애플 디바이스에 DNS 네임을 자동으로 부여하고 도메인 네이밍 서비스를 지원한다[4]. mDNS에서 각 노드는 자신의 DNS 서버를 운영하며 자신의 DNS 네임에 대한 쿼리(Query)를 송신하는 노드에게 IP 주소를 전달해준다. 봉주르에서 제공해주는 네이밍은 같은 로컬 네트워크(Local Network) 안에 사용 중인 기기들의 DNS 네임과 그에 대응하는 IPv4 또는 IPv6 주소를 특별한 보안키 없이 보여주기 때문에 프라이버시에 문제가 있다. 또한 mDNS는 멀티캐스트 기반으로 구성되어 있기 때문에 네트워크 상에 많은 트래픽을 발생시킨다. 그렇기 때문에 mDNS는 멀티홉으로 구성된 멀티 링크 서브넷과 Duty Cycling을 하는 센서로 구성되는 무선 센서 네트워크에서 디바이스를 위한 네이밍 서비스로는 적합하지 않다.

IPv6 네트워크 계층을 통해 사물인터넷 디바이스 DNS 네임 자동설정 및 네이밍 서비스 제공을 위해 DNSNA(DNS Name Autoconfiguration)가 제안되었다[5]. DNSNA는 IPv6의 이웃탐색 프로토콜(Neighbor Discovery, ND)[6]을 이용하여 IoT 디바이스를 위한 네이밍 서비스를 제공한다. ND의 라우터 광고(Router Advertisement, RA) 메시지는 DNS Search List(DNSSL)[7] 옵션을 통해 IoT 디바이스들에게 현재 위치한 링크에 해당되는 도메인 정보인 DNS Suffix를 제공한다. IoT 디바이스는 이렇게 전달받은 DNS Suffix에 디바이스 정보(예, Unique ID, 디바이스 모델)를 결합하여 DNS 네임을 생성한다. IoT 디바이스는 생성된 DNS 네임이 현재 링크에서 유일한지 검사한다. 이후에 현재 링크의 라우터는 Node Information(NI) 쿼리(Query)[8]를 통해 링크에 있는 IoT 디바이스의 DNS 네임과 대응하는 IPv6 주소를 수집한다. 라우터는 수집한 IoT 디바이스의 DNS 정보를 DNS 동적갱신(Dynamic Update)[9]을 통해 도메인을 관리하는 DNS 서버에 등록한다. 따라서 특정 네트워크 내의 IoT 디바이스의 DNS 정보는 DNS 서버에 등록되어 있기 때문에 어떤 노드가 다른 노드의 DNS 네임의 IPv6 주소를 알고 싶을 때는 유니캐스트 방식으로DNS 서버에게 쿼리(Query)를 전송한다.

미래에는 사물인터넷에 대한 관심과 활용이 많아질 것이다. 또한 현재에도 사물인터넷을 어떻게 효율적으로 관리할 수 있을 것인가에 대한 연구는 활발히 진행되고 있다. 따라서 본고에서는 사물인터넷에 대한 DNS 네임 서비스 기술에 대해 소개하려 한다. 본고의 본론에서는 사물인터넷을 위한 IoT 디바이스의 DNS 네이밍 서비스에 대한 선행연구와 프로토콜을 설명하고 보안적 이슈도 다루도록 한다. 본고의 결론에서는 본고에 정리 및 향후 연구 방향에 대해 기술하였다.

II. 본 론

1. 네이밍 서비스에 대한 연구

EPCglobal[10]은 상품코드의 국제표준 개발/관리 기구인 European Article Number(EAN)과 Unique Country Code(UCC)의 통합으로 탄생한 GS1이 2003년 11월에 설립한 자회사이다. Electronic Product Code(EPC)는 상품을 식별하는 코드이다. 특정 EPC에 대한 정보를 찾아 그 정보에 액세스 할 수 있도록 해주는 종합적인 서비스로 객체 네이밍 서비스(Object Naming Service, ONS)를 개발했다. ONS 검색 프로세스는 EPC 코드를 가지며, EPC의 객체정보를 가진 하나 이상의 네트워크에서 유일자원지시기(Uniform Resource Locator, URL)를 반환하여 정보를 제공한다. 그러나 ONS는 상품에만 제한적이기 때문에 IoT 디바이스를 위한 DNS 네임 레졸루션(Resolution) 서비스로는 적합하지 않다.

[11]의 논문 저자들은 이동 애드혹 네트워크(Mobile Ad-hoc Network, MANET) 환경에서 DNS 네이밍 서비스를 제안한다. 노드의 DNS 네임을 자동으로 설정하기 위해 네트워크 인터페이스의 Media Access Control(MAC) 주소를 EUI-64(64-bit Extended Unique Identifier)로 변환하여, 사용자 또는 디바이스 ID와 MANET을 위한 DNS 서픽스(예, adhoc)를 이용해 DNS 네임을 자동생성 및 유일성을 검증한다. 하지만 EUI-64 기반의 DNS 네임은 일반 사용자들이 이해하기 어렵고, 다른 디바이스들이 이러한 DNS 네임을 통해 유용한 정보를 얻을 수 없다. 따라서 이 기법은 IoT DNS 네이밍 서비스로는 적합하지 않다.

2. Mutlicat DNS(mDNS)

애플에서 개발한 봉주르는 DNS 네이밍 및 서비스 디스커버리를 제공하는 제로콘프 소프트웨어이다[4]. 봉주르는 애플 디바이스의 등록과 검색 서비스를 제공한다. 봉주르는 mDNS[12]을 이용하여 로컬 네트워크에 참여한 기기들의 DNS 네임을 기기 또는 사용자 ID와 그 네트워크의 DNS Suffix를 기반으로 자동생성하여 DNS 네이밍 서비스를 제공한다. 하지만 mDNS는 기본적으로 멀티캐스트 기반으로 네임 레졸루션을 수행하므로 네트워크에 많은 트래픽을 발생시킨다. 이것은 멀티홉으로 구성된 멀티링크 서브넷(Multi-link Subnet) 같은 네트워크에서는 적절하지 않다. 또한 인터넷에 연결된 디바이스가 항상 동작하는 경우에만 DNS 네임 요청 메시지에 응답할 수 있고 에너지 절약을 위해 슬립모드를 지원하는 센서와 같은 IoT 디바이스에 사용되기에는 부적합하다. 또한 제공하는 DNS 네임이 평문(Plaintext)으로 공개되므로 디바이스의 정보가 쉽게 노출된다.

봉주르는 또한 DNS-based Service Discovery(DNS-SD)[13]를 이용해서 특정 서비스 타입을 지원하는 인스턴스(Instance) 목록을 검색한다. DNS-SD는 DNS SRV Record[23]와 DNS TXT Record[24]를 이용하여 특정 서비스 인스턴스를 기술한다. DNS SRV Record는 〈Instance〉.〈Service〉.〈Domain〉 형식의 네임을 가지며 Service의 Instance를 실행하는 타겟 호스트와 포트 정보를 포함한다. DNS TXT Record는 Service의 Instance에 대한 부가적인 정보를 제공한다. DNS-SD 클라이언트는 〈Service〉.〈Domain〉의 네임을 가지고 DNS PTR Record[24]를 통해 주어진 Service에 대한 가용한 Instance들의 목록을 발견하고, 그 목록에 있는 Instance를 통해 원하는 서비스를 제공받는다.

3. DNSNA

가. DNSNA 시스템 구성

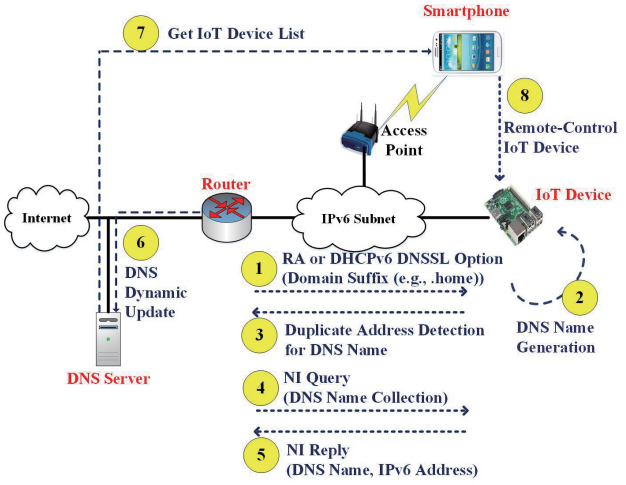


그림 2. DNSNA 시스템 구성도

DNSNA는 홈 네트워크 및 도로 네트워크와 같은 다양한 IoT 환경에서 DNS 네임 생성과 레졸루션을 위해 〈그림 2〉와 같은 새로운 아키텍쳐를 제시한다. DNSNA는 기존의 인터넷 DNS 서비스와 같이 사용자 단말 IPv6 호스트(예, 스마트폰)는 IPv6 호스트인 IoT 디바이스의 DNS 네임의 IPv6 주소의 레졸루션을 위해 〈그림 2〉의 DNS 서버와의 유니캐스트 통신을 통해 수행한다. 이러한 네임 레졸루션은 IoT 디바이스에 대해 효율적인 DNS 네임 서비스를 제공한다. 또한 DNSNA는 디바이스 자신의 카테고리 및 모델 이름을 이용해서 쉽게 자신의 DNS 네임을 설정할 수 있다. DNSNA는 RA[7] 또는 Dynamic Host Configuration Protocol Version 6(DHCPv6)[14]를 통해 DNSSL을 얻으며, 획득된 Domain Suffix와 IoT 디바이스의 정보를 통해 IoT DNS 네임을 생성 및 네임의 유일성 검사를 수행하기 위해 IPv6 ND를 사용한다.

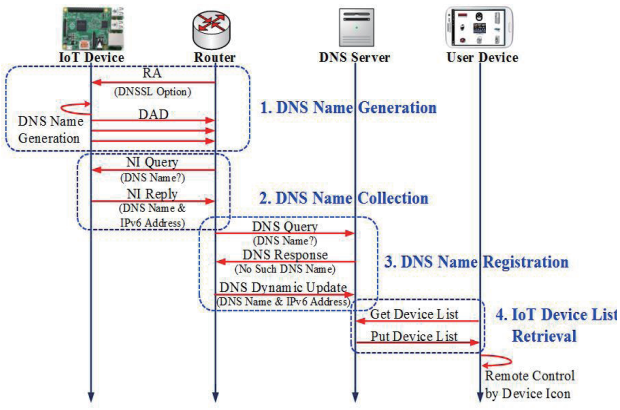


그림 3. DNSNA 시퀀스 다이어그램

〈그림 3〉은 DNSNA의 시퀀스(Sequence) 다이어그램이다. DNS 네임 생성, DNS 네임 수집, DNS 네임 등록, DNS 네임 획득 및 조작으로 총 4단계로 구분한다.

첫 번째 DNS 네임 생성 단계에서는 라우터가 RA 메시지의 DNSSL 옵션을 보내는 것으로 진행된다. IoT 디바이스는 DNSSL 옵션의 유효성을 검사한 후에 자신의 DNS 네임을 생성하고 중복주소검출(Duplicate Address Detection, DAD)을 위한 메시지를 송신한다. DAD 메시지에 대한 응답이 없으면 중복되지 않았다고 판단하고 DNS 네임을 생성한다.

두 번째 DNS 네임 수집 단계는 다음과 같다. 라우터는 각 IoT 디바이스의 DNS 네임과 이에 상응하는 IPv6 주소를 수집하기 위해 IoT 디바이스들에게 멀티캐스트로 NI 쿼리(Query)를 송신한다. 각 IoT 디바이스는 해당 쿼리(Query)에 대한 응답으로 NI Reply로 자신의 DNS 네임 및 IPv6 주소를 라우터에게 송신한다.

세 번째 DNS 네임 등록 단계는 다음과 같다. 라우터는 각 IoT 디바이스의 DNS 네임을 DNS 서버에 등록하기 위해 NI Reply를 통해 받은 DNS 네임이 이미 등록되었는지 DNS 쿼리를 통해 확인한다. DNS 서버에 등록되어 있지 않으면 DNS 서버는 IoT 디바이스의 DNS 네임 및 IPv6 주소를 DNS 동적갱신을 통해 등록한다.

마지막으로 DNS 네임 획득 및 조작 단계는 사용자가 각 IoT 디바이스를 조작하기 위해 스마트폰 또는 태블릿 PC와 같은 조작가능 디바이스로 DNS 서버에 접속하여 IoT 디바이스들의 DNS 네임 및 IPv6 주소를 획득하여 디바이스를 발견하고, 각 디바이스가 제공하는 서비스를 원격에서 요청할 수 있다.

나. DNS 네임 포맷

〈그림 4〉는 디바이스 정보를 활용한 DNS 네임 포맷이다. 포맷에 대한 설명은 그림 아래에 있다.

unique_id.device_model.device_category.location.domain_name

그림 4. 디바이스 정보를 활용한 DNS 네임 포맷

- unique_id: ASCII 문자로 DNS 네임의 유일성을 보장하기 위한 유일한 식별자. 식별자는 일련번호 또는 제품명과 같이 가독성이 있는 영어 알파벳과 숫자로 구성될 수 있다.
- device_model: ASCII 문자로 디바이스의 모델 이름. 디바이스 제작 시 제조업체로부터 부여받는다.
- device_category: ASCII 문자로 디바이스의 카테고리. TV, 스마트 카, 태블릿 등과 같음.
- location: 디바이스의 물리적인 위치. 미시적 위치(예, 중앙, 코너, 벽 등)와 거시적인 위치(예, 거실, 주방, 침실 등)로 구성되어 있다. 이 필드는 선택적이므로 DNS 네임 포맷에서 제외될 수 있다.
- domain_name: RFC 3315의 도메인 네임의 표현에 따라 인코딩된 DNS 도메인 네임[15].

〈그림 5〉는 oneM2M에서 제안한 Object Identifier(OID)를 응용한 DNS 네임 포맷이다[16]. OID는 oneM2M에서 제안한 것으로 사물인터넷에서 디바이스 및 서비스와 같은 다양한 객체를 식별하기 위한 객체 식별자(OID)를 정의한다.

unique_id.object_identifier.location.domain_name

그림 5. Object Identifier를 이용한 DNS 네임 포맷

- unique_id, location, domain_name: 〈그림 3〉의 필드 설명과 동일하다.
- object_identifier: 디바이스의 오브젝트 식별자로써 oneM2M에서 정의된 상위 아크(higher arc), 즉 M2M 노드 지시 ID(즉 관리 기관, 관청, 데이터 국가 코드, M2M 노드의 결합 문자열), 일련의 4개의 아크(즉 제조사 ID, 모델 ID, 시리얼 ID, 확장 ID)로 구성된다. 각 필드는 언더스코어 문자 '_'로 구분된다[16].

다. DNS 네임 생성

DNS 네임 자동설정은 RA DNSSL 옵션[7] 또는 DHCPv6 DNSSL 옵션[14]을 통해 수신한 DNS 서픽스(Suffix) 기반으로 수행된다. IPv6 호스트로서 디바이스는 DNSSL 옵션을 수신할 때 디바이스는 DNSSL 옵션의 유효성을 검사한다. 옵션이 유효하면 DNSSL 옵션에 있는 각각의 DNS 서픽스 도메인 네임에 대해 다음과 같은 과정으로 DNS 네임 자동설정을 수행한다.

1) 호스트는 수신한 DNS 서픽스 도메인 네임, 디바이스 정보(제조사 ID, 모델 ID, 시리얼 ID) 및 유일하다고 판단된 식별자(즉, unique_id)로 자신의 DNS 네임을 생성한다.
2) 호스트는 64 비트 네트워크 프리픽스 및 상기 DNS 이름의 MD5 해시 값의 마지막 64 비트를 가지고 임시 주소인 IPv6 유니캐스트 주소를 구성한다.
3) 호스트는 임시 IPv6 주소에 해당하는 요청 노드 멀티캐스트 주소(Solicited-Node Multicast Address)를 생성한다[6].
4) 호스트는 요청 노드 멀티캐스트 주소와 함께 IPv6 주소에 대한 중복주소검사(DAD)를 수행한다[6][17].
5) 만약 DAD에 대한 응답을 받지 않으면, 호스트는 이 임시 IPv6 주소를 자신의 IPv6 유니캐스트 주소로 설정하고, 생성된 DNS 네임을 로컬 상에서 유일한 것으로 간주한다. 반면에, DAD가 실패하면 DNS 네임이 충돌했기 때문이므로 1번으로 다시 돌아가서 또 다른 유일한 식별자를 선택하여 새로운 DNS 네임을 생성한다.
6) DNS 네임이 유일하다고 입증되면, 이 DNS 네임을 디바이스 네임으로 설정하고 주어진 DNS 서픽스 도메인 네임에 대한 DNS 자동설정을 완료한다. 또한, 호스트는 다른 호스트로부터 자신의 DNS 이름이 사용되는 것을 방지하기 위해 DNS 이름에 해당하는 요청 노드 멀티캐스트 주소 그룹에 조인한다.

DNS Search List가 하나보다 많은 DNS 서픽스 도메인 네임을 포함하고 있을 때는 IPv6 호스트는 모든 DNS 서픽스가 DNS 네임 자동설정에 사용될 때까지 위의 과정을 반복한다.

라. DNS 네임 수집

IPv6 호스트들로서 디바이스들은 자신의 DNS 네임을 성공적으로 자동설정 한 이후에 동일한 서브넷에 있는 라우터는 수집 노드로서 IPv6 노드 정보(NI) 프로토콜[8]을 사용하여 디바이스 DNS 네임을 수집한다.

수집 노드가 사전에 다른 디바이스에 대한 정보 없이 그 디바이스들의 DNS 네임과 IPv6 주소를 수집하기 위해서는 새로운 NI 쿼리가 정의되어야 한다. 즉, 새로운 ICMPv6 코드(예, 3)가 IPv6 호스트 DNS 네임의 수집을 위해 정의되어야 한다. NI 쿼리는 동일한 서브넷에 있는 모든 IPv6 호스트에 대한 것이므로 NI 쿼리의 데이터 필드 없는 ICMPv6 메시지로 생성된다. NI 타입에 대한 Qtype 필드는 노드 네임을 위해 2로 설정된다.

(a) 텍스트 기반 디바이스 탐색 (b) 텍스트 기반 서비스 탐색 (c) 그래픽 기반 디바이스 탐색

그림 6. DNS 네이밍 서비스 기반의 IoT 디바이스 및 서비스 탐색

쿼리(Query)는 수집 노드에 의해 링크로컬 멀티캐스트 주소로 송신되어야 한다. 하나의 링크로컬 멀티캐스트 주소(예, 모든 노드 멀티캐스트 주소, FF02::1)는 디바이스 DNS 네임 수집을 위해 정의되는 것을 가정한다. 또한, 모든 IPv6 호스트(즉, IoT 디바이스)는 디바이스 DNS 네임 수집에 응하기 위해 이러한 멀티캐스트 주소에 조인하고 있다고 가정한다.

IPv6 호스트가 수집 노드에 의해 멀티캐스트로 송신된 쿼리를 수신하면 그 호스트는 멀티캐스트 수신자 탐색 버전 2(Multicast Listener Discovery Version 2, MLDv2)[18]에 정의된 것과 같이 0초에서 쿼리 응답 시간 간격 사이의 랜덤한 시간 간격으로 응답 메시지를 송신한다. 이러한 랜덤한 시간 간격의 응답은 IoT 디바이스들의 응답 시간을 분산시킴으로써 작은 프레임 충돌 확률로 디바이스 DNS 네임을 수집하기 위해서이다.

DNS 네임 레졸루션 서비스를 위해 수집 노드는 각 IPv6 호스트에 대한 DNS 네임과 IPv6 주소의 쌍을 DNS 네임의 DNS 도메인 서픽스를 관리하는 전담 DNS 서버(Designated DNS Server)에 등록한다. 이것은 수집 노드가 지정된 DNS 서버 안에 DNS 네임들을 안전하게 등록하기 위해 그 지정된 DNS 서버는 DNS 보안을 위한 DNSSEC[19][20]을 사용할 수 있다. 이 DNS 네임과 IPv6 주소 쌍의 등록은 DNS 동적갱신을 통해 수행될 수 있다. 지정된 DNS 서버 안에 DNS 네임을 등록하기 전에, 수집 노드는 DNS 네임 레졸루션 서비스를 위한 DNS 쿼리를 전송하여 등록될 DNS 이름이 유일한지 확인해야 한다. 만약 쿼리된 DNS 네임에 해당하는 IPv6 주소가 없는 경우, 수집 노드는 DNS 네임과 IPv6 주소를 지정된 DNS 서버에 등록한다. 반면에 해당하는 IPv6 주소가 있다면, DNS 네임이 중복되었다 판단하고 수집 노드는 NI 프로토콜을 이용하여 해당 IoT 디바이스인 IPv6 호스트에게 DNS 네임 중복에 대한 에러 메시지와 중복된 DNS 네임을 함께 전달한다. IoT 디바이스가 이러한 DNS 네임에 대한 중복 에러를 받으면, 그 IoT 디바이스는 새로운 DNS 네임을 생성해야 하므로 자신의 식별자를 변경한 뒤 디바이스의 DNS 네임 생성 과정을 반복한다.

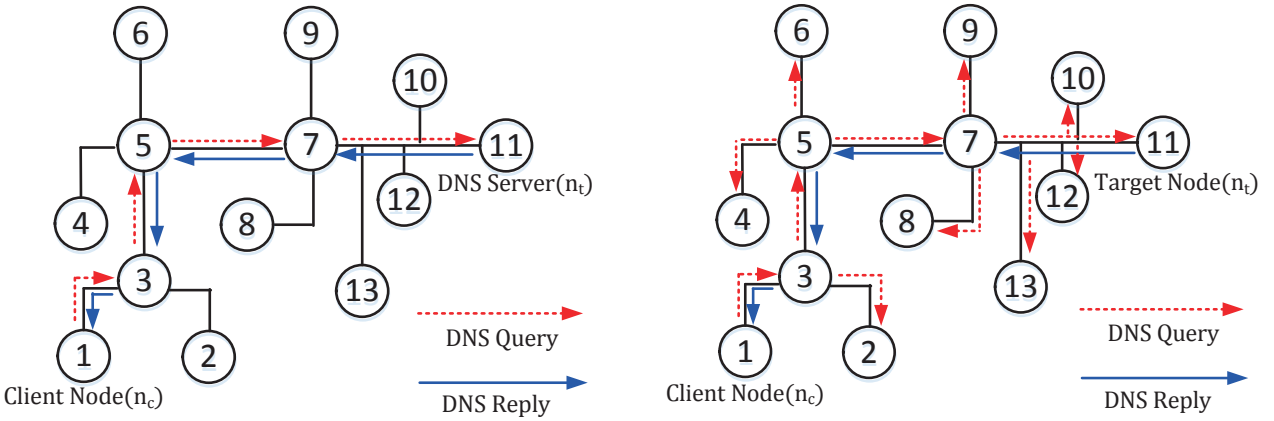
마. DNS 네임 획득 및 조작

스마트폰 또는 태블릿 PC와 같은 스마트 디바이스는 IoT 디바이스 DNS 네임을 관리하는 로컬 DNS 서버에 접속하여 IoT 디바이스의 DNS 네임들을 획득할 수 있다. 스마트 디바이스는 DNS 네임을 포함하는 DNS 존 파일(Zone File)을 획득할 수 있다면 홈 네트워크와 같은 로컬 네트워크에 있는 IoT 디바이스의 정보를 사용자에게 보여줄 수 있다. 이러한 IoT 다바이스 정보를 가지고 사용자는 원격에서 IoT 디바이스를 모니터링 및 제어를 할 수 있다.

〈그림 6(a)〉는 텍스트 기반 디바이스 탐색 서비스로서 태블릿 PC를 통해 댁내에 배치된 IoT 디바이스들의 종류를 보이고 있다. 이것을 클릭하면 〈그림 6(b)〉와 같은 텍스트 형태로 서비스 정보를 볼 수 있다. 〈그림 6(c)〉는 그래픽 기반 디바이스 탐색 서비스를 보여주고 있는데, 보다 사용자에게 친근한 사용자 인터페이스를 제공할 수 있다. 이러한 그래픽 기반 디바이스 탐색 서비스을 위해서는 IoT 디바이스의 위치정보를 파악하는 위치측정(Localication) 서비스(예, SALA[21])가 기반이 되어야 한다.

바. mDNS와 DNSNA의 성능 분석

이 절에서는 mDNS와 DNSNA의 성능에 대해 분석하는 내용을 다루도록 한다. 〈그림 7〉은 멀티링크 서브넷에서의 DNSNA와 mDNS의 DNS 네임 레졸루션을 위한 트래픽량을 비교 및 분석한 그림이다. DNSNA는 〈그림 7(a)〉와 같이 유니캐스트 기반으로 레졸루션을 수행하므로 DNS 쿼리 메시지의 전송수는 쿼리를 생성하는 클라이언트 노드와 DNS 서버사이의 홉(Hop) 수에 의해 결정된다. 반면 mDNS는 〈그림 7(b)〉와 같이 전체 네트워크에서 멀티캐스트에 의해 레졸루션을 수행하므로 DNS



(a) DNSNA의 DNS 네임 레졸루션 (b) mDNS의 DNS 네임 레졸루션

그림 7. 다중링크 네트워크 환경에서 DNSNA기법과 mDNS기법의 트래픽 비교

쿼리 메시지의 전송수는 네트워크의 총링크수에 의해 결정된다. DNSNA의 경우 노드에서 DNS 서버까지의 거리가 네트워크의 지름인 m 홉(Hop)이라고 가정한다면 최악의 경우(Worst Case)는 DNS 쿼리가 2*m홉만큼 포워딩 될 것이다. 이것은 메시지 복잡도로 $O(m)$를 의미한다. 반면 mDNS의 경우 〈그림 7(b)〉와 같이 멀티캐스트 되므로 최악의 경우는 DNS 쿼리가 네트워크 전체 링크로 멀티캐스트 되므로 메시지 복잡도는 링크 개수인 $O(|E|)$이다.

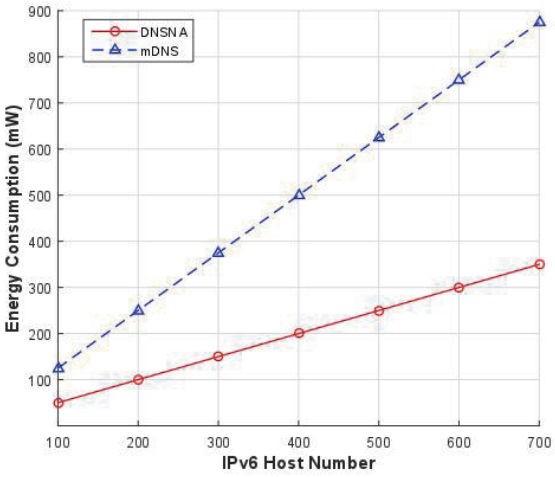


그림 8. DNS 네임 레졸루션에 대한 에너지 소비

〈그림 8〉은 DNSNA와 mDNS의 DNS 네임 레졸루션에 대한 에너지 소비량 비교를 Matlab으로 표현한 그림이다. 실험에 대해 에너지 소비는 1 밀리와트(milliwatt) 당 1 메시지가 소비 되었다고 가정하였다[5]. IPv6 호스트수가 증가할수록 두 방법 모두 더 많은 에너지를 소비하였다. 그러나 〈그림 8〉과 같이 DNSNA가 mDNS보다 적은 에너지를 소비하는 것을 볼 수 있다. 이것은 DNSNA가 유니캐스트 기반으로 동작하기 때문이다. 마지막 아래 〈표 1〉은 DNSNA와 mDNS에 대해 비교한 것을 표로 정리한 것이다.

표 1. DNSNA와 mDNS의 비교표

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

4. Secure DNSNA

DNSNA의 모든 메시지들은 평문(Plaintext)으로 보내지기 때문에 스니핑(Sniffing) 공격에 취약하다는 단점이 있다. 그래서 악의적인 목적을 가진 해커는 스니핑을 사용해서 유저의 정보를 얻을 수 있을 것이다. 또한 악의적인 목적을 가진 IoT 디바이스는 사용자의 네트워크에 Access Point(AP)나 이더넷(Ethernet) 네트워크를 통해 접속하여 DNSNA 프로토콜을 통해 디바이스의 DNS 네임과 IPv6 주소를 쉽게 등록할 수 있다.

왜냐하면 DNSNA는 어떠한 인증과정이 없이 IoT 디바이스의 DNS 네임을 등록할 수 있기 때문이다.

만약 악의적인 목적을 가진 IoT 디바이스가 AP를 통해 DNS 서버로부터 서비스를 제공 받고 DNS 네임을 받는다면, 해커는 AP가 구성하는 무선랜에 연결된 디바이스들의 정보들을 쉽게 얻을 수 있을 것이다. 이와 같이 인증과정이 없는 DNS 네임 등록은 모든 디바이스들을 해커로부터 보안 공격을 받을 수 있게 노출시킨다. 따라서 다양한 보안 위협으로부터 정보를 보호하기 위해 보안이 추가된 모델인 Secure DNSNA(SDNSNA)가 제안되었다[22].

가. SDNSNA 시스템 구성

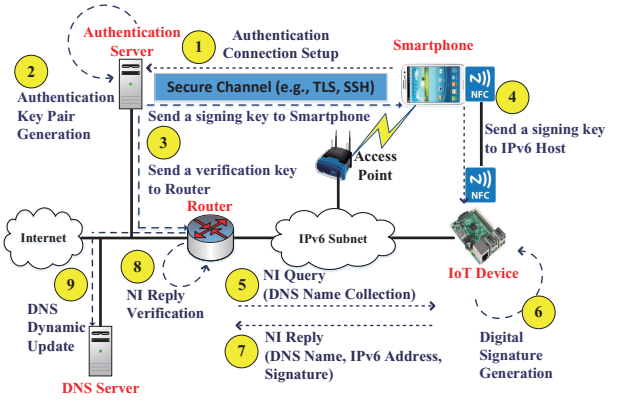


그림 9. SDNSNA 시스템 구성도

〈그림 9〉는 SDNSNA 시스템 구성도를 보여주고 있다. IoT 디바이스의 DNS 네임을 DNS 서버에 등록하는 부분은 이전에 설명했던 3장의 DNSNA의 프로토콜을 이용한다. 보안성을 제공하는 DNS 등록을 위한 SDNSNA는 DNSNA 절차에 스마트폰 기반의 IoT 디바이스 인증 절차를 선행적으로 수행한다. SDNSNA의 IoT 디바이스 인증은 〈그림 9〉와 같이 스마트폰이 인증서버(Authentication Server, AS)로부터 수신한 서명키(Signing Key)를 IoT 디바이스(즉, IPv6 Host)와의 Near Field Communication(NFC) 통신을 통해 물리적으로 인증하는 것이다. 수 센티미터의 통신범위를 갖는 NFC를 통신 매체로 이용하는 이유는 스마트폰이 보다 긴 통신 범위를 갖는 WiFi나 Bluetooth로 IoT 디바이스에게 서명키를 전송할 때 무선 신호의 전파 특성에 의해 서명키가 주변 디바이스나 컴퓨터로 노출되는 것을 방지할 수 있기 때문이다. SDNSNA에서의 인증은 비대칭키 기반의 전자서명을 통해 수행한다.

나. SDNSNA 동작 과정

SDNSNA를 이용한 IoT 디바이스의 DNS 정보의 등록은 〈그림 9〉와 같이 9개의 스텝을 통해 수행된다. 〈그림 9〉의 스텝 1에서 스마트폰은 인증서버(AS)와 인증을 위한 보안 채널(예, TLS 또는 SSH)을 형성한다. 스텝 2에서는 AS는 인증을 위한 두 개의 Key를 생성한다. 첫 번째 Key는 IoT 디바이스가 DNS 등록 메시지에 대한 전자서명을 생성하기 위한 서명키(Signing Key)이고, 두 번째 Key는 라우터가 DNS 등록 메시지의 전자서명을 검증하는 검증키(Verification Key)이다. 스텝 3에서는 AS는 서명키를 스마트폰에게 전송하고, 검증키를 라우터에게 전송한다. 스텝 4에서는 스마트폰은 인증할 IoT 디바이스에게 NFC 통신을 통해 서명키를 전달한다. 스텝 5에서 라우터가 IoT 디바이스의 DNS 네임 수집을 위한 NI Query를 IPv6 Subnet에 멀티캐스트한다. 스텝 6을 수행하기에 앞서서 IPv6 Host인 IoT 디바이스는 〈그림 2〉에서 RA 또는 DHCPv6 DNSSL 옵션에 포함된 Domain Suffix와 자신의 디바이스 정보 기반으로 〈그림 4〉의 DNS 포맷에 따라 자신의 DNS 네임과 이에 해당하는 IPv6 주소를 생성하여 DNS 네임 검증과정을 통해 자신의 로컬 스토리지(Local Storage)에 저장한다. 스텝 6에서 DNS 네임 정보 등록을 위한 NI Reply 메시지를 생성하여 자신의 서명키로 전자서명하여 Signature를 NI Reply 메시지에 포함시킨다. 스텝 7에서 IoT 디바이스는 NI Reply를 라우터에게 전송한다. 스텝 8에서 라우터는 자신의 검증키로 NI Reply를 검증한다. 즉, 라우터는 NI Reply를 받았을 때, NI Reply 안에 서명정보가 있으면 AS로부터 받았던 검증키를 이용해서 NI Reply를 검증한다. 만약 이 과정을 통해 해당 NI Reply가 유효하다고 판단되면 스텝 9와 같이 라우터는 DNS 네임 정보를 등록하기 위해 DNS 동적갱신을 통해 DNS 네임 정보를 DNS 서버에 등록한다[9].

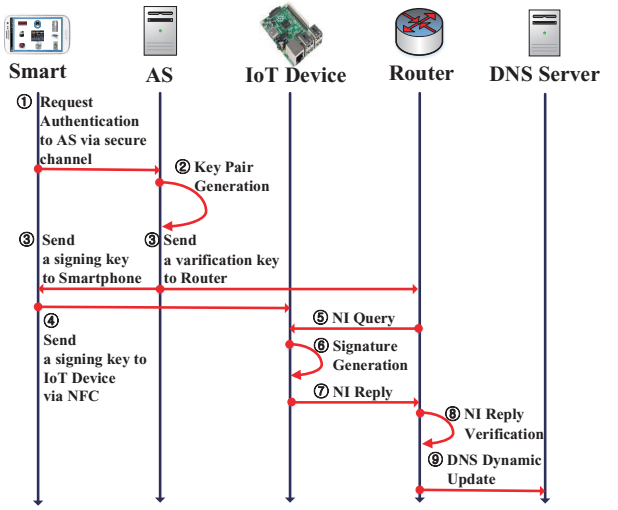


그림 10. SDNSNA 시퀀스 다이어그램

〈그림 10〉은 〈그림 9〉에 대한 IoT 디바이스 DNS 정보 등록 과정에 대한 SDNSNA 시퀀스 다이어그램을 보여주고 있다.

III. 결 론

기하급수적으로 증가하는 IoT 디바이스를 포함하는 인터넷 시대에서 IoT 디바이스의 효율적인 관리가 필요하다. 본고에서는 IoT 디바이스를 관리하기 위한 IoT 디바이스의 DNS 네임을 자동설정, 자동등록 및 네임 레졸루션하는 DNS 네이밍 기술에 대한 동향을 살펴보았다. DNSNA는 이러한 IoT 디바이스의 네이밍 서비스를 위한 효과적인 시스템을 제시하고 있고, SDNSNA는 DNSNA에 IoT 디바이스의 인증 절차를 포함한 시스템을 제시한다. 향후 연구주제로써 수많은 IoT 디바이스들에 대한 동시다발적인 DNS 네임 레졸루션 요청을 효과적으로 처리하기 위해서는 확장성을 고려한 분산 DNS 시스템(Scalable Distributed DNS Systems)에 대한 연구가 필요할 전망이다.

참 고 문 헌

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions", Future Generation Computer Systems, Vol. 29, No. 7, pp. 1645-1660, Sep. 2013.

[2] Gartner's 2014 Hype Cycle [Online]. Available: http://www.gartner.com/newsroom/id/2819918

[3] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", IETF RFC 2460, Dec. 1998.

[4] Apple Bonjour [Online]. Available: https://developer.apple.com/bonjour

[5] Sejun Lee, Jaehoon (Paul) Jeong, and Jung-Soo Park, "DNSNA: DNS Name Autoconfiguration for Internet of Things Devices", The 18th International Conference on Advanced Communications Technology (ICACT 2016), Outstanding Paper, Phoenix, Korea, Jan. 2016.

[6] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", IETF RFC 4861, Sep. 2007.

[7] J. Jeong, S. Park, L. Beloeil, and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", IETF RFC 6106, Nov. 2010.

[8] M. Crawford and B. Haberman, "IPv6 Node Information Queries", IETF RFC 4620, Aug. 2006.

[9] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", IETF RFC 2136, Apr. 1997.

[10] GS1, "EPCglobal", http://www.gs1.org/ epcglobal.

[11] 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.

[12] S. Cheshire and M. Krochmal, "Multicast DNS", IETF RFC 6762, Feb. 2013.

[13] S. Cheshire and M. Krochmal, "DNS-Based Service Discovery", IETF RFC 6763, Feb. 2013.

[14] R. Droms, "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", IETF RFC 3646, Dec. 2003.

[15] R. Droms, J. Bound, B. Volz, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", IETF RFC 3315, Jun. 2003.

[16] M2M, "Object Identifier based M2M Device Identification Scheme", http://www.onem2m.org.

[17] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, Sep. 2007.

[18] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[19] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, Mar. 2005.

[20] S. Weiler and D. Blacka, "Clarifications and Implementation Notes for DNS Security (DNSSEC)", RFC 6840, Feb. 2013.

[21] Jaehoon (Paul) Jeong, Solchan Yeon, Taemoon Kim, Hyunsoo Lee, Song Min Kim, and Sang-Chul Kim, "SALA: Smartphone-Assisted Localization Algorithm for Positioning Indoor IoT Devices", Springer Wireless Networks, Jun. 2016.

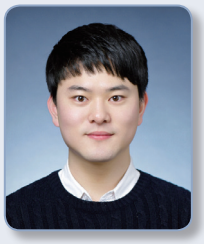
[22] Keuntae Lee, Hyungsuk Kang, Jeahoon (Paul)

Jeong, Hyoungshick Kim, and Jung-Soo Park, "Secure DNS Name Autoconfiguration for IPv6 Internet-of-Things Devices", International Conference on ICT Convergence (ICTC 2016), Jeju, Korea, Oct. 2016.

[23] A. Gulbrandsen, P. Vixie, and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", IETF RFC 2782, Feb. 2000.

[24] P. Mockapetris, "Domain Names - Implementation and Specification", IETF RFC 1035, Nov. 1987.

약 력



이 근 태

2009년~2016년 경희대학교 컴퓨터공학 공학사
2016년~현재 성균관대학교 전자전기컴퓨터공학 공학석사과정
관심분야: Internet-of-Things (IoT), Wireless Sensor Networks, Software-Defined Networking (SDN), Network Functions Virtualization (NFV)

정 재 훈

1992년~1999년 성균관대학교 공학사
1999년~2001년 서울대학교 공학석사
2004년~2009년 미네소타대학교 공학박사
2001년~2004년 한국전자통신연구원 표준연구센터 연구원
2010년~2012년 Brocade Communications Systems 소프트웨어 엔지니어
2012년~현재 성균관대학교 정보통신대학 소프트웨어학과 조교수
관심분야: Internet of Things (IoT), Security Services for Software-Defined Networking (SDN) and Network Functions Virtualization (NFV), Cyber-Physical Systems (CPS), Indoor/Outdoor Navigation, Vehicular Ad Hoc Networks, Wireless Sensor Networks, Mobile Ad Hoc Networks