NAT 환경에서 실시간 사물인터넷 서비스 지원을 위한 사물인터넷 마이크로 서비스 구조 설계

김근수, 김소용, 김민지, 고석주*
경북대학교 컴퓨터학부

kks36145919@gmail.com, *sjkoh@knu.ac.kr

Design of Internet-of-Things Microservices Architecture for Realtime Messaging in Network Address Translation (NAT)

Keun-Soo Kim, So-Yong Kim, Min-Ji Kim, Seok-Joo Koh*
Kyungpook National Univ.

요 약

스마트 홈, 스마트 시티, 스마트 팩토리 등 다양한 분야와 융합하여 개발되고 있는 사물인터넷 서비스는 매우 빠르게 발전하고 있다. 그 중 일부 서비스는 좁은 영역에서 제공되며, 이러한 서비스들은 NAT 기술과 함께 서비스가 제공된다. 하지만 NAT 는 외부에 있는 노드가 내부에 있는 노드에 먼저 메시지를 전송할 수 없는 문제를 가지며, 이는 서버에서 클라이언트로 제어메시지가 빈번히 전송될 필요가 있는 사물인터넷 서비스 구조에 적합하지 않다. 본 논문에서는 구조적으로 해당 문제를 해결하기 위해 설계한 사물인터넷 마이크로 서비스 구조를 제안하며, 간단한 테스트베드를 통해 서비스 구조를 검증한다.

Ⅰ. 서 론

최근 원격 제어, 상호작용을 통한 자동화까지 매우 다양한 사물인터넷 서비스들이 활발히 개발되고 있다. 사물인터넷 서비스는 4 개의 계층(Sensing Layer, Network Layer, Middle-ware Layer, Application Layer)으로 구성되며, 스마트 홈, 스마트 시티, 스마트 팩토리 등 매우 다양한 서비스에 사용된다. 그 중 좁은 영역에서 제공되는 일부 스마트 홈, 스마트 팩토리 서비스에는 NAT 기술이 사용된다.

NAT 기술은 기존 IPv4(Internet Protocol version 4)주소가 고갈되어 감에 따라 이를 해결하는 방안으로 개발된 기술이다. 부족한 공인 IP 주소를 여러 개의 사설 IP 주소로 변환 및 할당하여 더 많은 클라이언트가 사용할 수 있도록 한다.

하지만, 이러한 NAT 는 2 가지 문제를 유발할 수 있다. 첫 번째로 NAT 외부 노드에서 내부 노드로의 통신 시작이 불가능하다. 예를 들어, 공인 IP 를 사용하는 서버가 NAT 내부의 사설 IP 를 사용하는 클라이언트를 발견할 수 없어 외부에서 통신 시작이 불가능하다. 두 번째로 NAT 내부에서 외부로 통신을 시작하여 NAT 바인딩이 수행되어도 NAT 매핑 테이블의 타임아웃 시간동안 통신을 하지 않으면 NAT 장비는 통신이 종료되었다고 판단하여 세션 정보를 제거한다. 따라서 NAT 외부 노드는 내부 노드와 통신이 불가하고 실시간 통신이 지속되지 못하는 문제가 있다.

또한, NAT 환경에서 실시간 메시지 교환을 위한 사물인터넷 서비스 아키텍처에 관한 연구는 미비한 수준이다. 따라서 본 논문에서는 NAT 환경에서 실시간 메시지 교환을 지원하는 사물인터넷 마이크로 서비스 아키텍처를 제안한다.

2 장에서는 사물인터넷 서비스에 사용되는 주요 프로토콜 CoAP, MQTT 소개와 함께 마이크로 서비스 아키텍처를 관련 연구로 소개한다. 3 장에서는 NAT 환경에서 실시간 메시지 교환을 위한 서비스 아키텍처를 제안하고, 4 장에서 검증한 후 5 장에서 결론을 맺는다.

Ⅱ. 관련 연구

CoAP (Constrained Application Protocol)[1] 은 HTTP 와 같은 RESTful Architecture 를 제공하는 응용 프로토콜이다. RESTful Architecture 덕분에 웹 서비스와 호환성이 좋으며, 신뢰성 있는 데이터 전송 등 UDP 가 지원하지 않는 서비스 개발에 필수적인 기능을 제공하여 많은 사물인터넷 서비스 개발에 사용되고 있다.

MQTT 는 IBM 이 M2M 및 사물인터넷 환경을 고려하여 개발한 Publish-Subscribe 구조의 경량화 메시지 전송 프로토콜이다[2]. HTTP 등 기존 프로토콜과 비교하여 작은 고정 헤더를 가지며, 적은 오버헤드로 패킷을 전송할 수 있으므로 낮은 대역폭 환경에서 효율적이다.

마이크로 서비스는 애플리케이션 구축을 위한 아키텍처 기반의 접근 방식이다. 마이크로 서비스는 모놀리식(Monolithic) 접근 방식과 달리 애플리케이션을 핵심 기능으로 세분화된 모듈로 구성된다.

애플리케이션 개발 초기에는 전체 애플리케이션 소스 코드가 하나의 배포 유닛으로 내장된 모놀리식 형태로 구성되었다. 이러한 형태에서는 QA (Quality Assurance) 주기에 따라 대규모 업데이트를 수행해야 하였으며, 일부 애플리케이션의 업데이트로 인해 오류가 발생할 경우, 전체를 오프라인으로 전환하고 운영 규모를 축소시킨 다음 문제를 해결해야했다.

모놀리식 형태의 단점을 극복하기 위해 최근 서비스들은 마이크로 서비스 아키텍처로 개발되고 있다. 마이크로 서비스란 애플리케이션을 여러 개의 다른 역할을 수행하는 모듈로 분리하였을 때 각 모듈을 의미하며, 이렇게 마이크로 서비스를 분리하여 여러 개의 작은 애플리케이션으로 쪼개어 변경과 조합이 가능하도록 만든 아키텍처를 마이크로 서비스 아키텍처라 한다. 소규모의 서로 상호 보완적인

서비스들을 묶어 큰 서비스를 완성하는 것이 마이크로 서비스 아키텍처의 핵심이다. 그림 1 은 모놀리식 구조와 마이크로 서비스 구조의 비교를 보여준다.

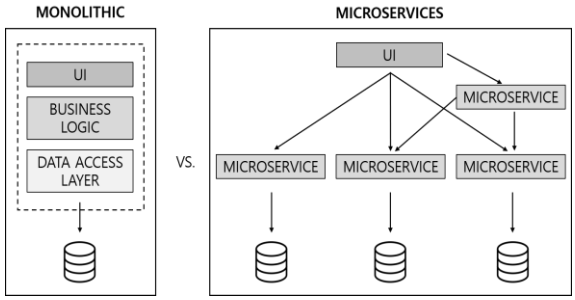

그림 1. 모놀리식 및 마이크로 서비스 구조

Ⅲ. 제안 기법

본 논문에서는 NAT 환경에서 실시간 메시지 교환을 위한 사물인터넷 마이크로 서비스 아키텍처 (IoT-RMSA, Internet of Things Realtime Messaging Microservice Architecture)를 제안한다. IoT-RMSA 는 크게 4 개의 구성요소 (IoT Server, Request Broker, Gateway, Device and Actuator) 로 구성된다. 그림 2 는 IoT-RMSA 의 구조를 보여준다.

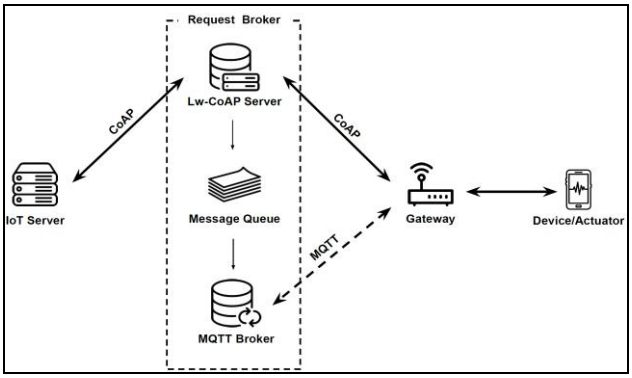

그림 2. 제안 기법 개요

제안 구조인 IoT-RMSA 의 핵심인 Request Broker 는 IoT Server 가 Gateway 에게 전달할 요청 메시지를 관리하는 역할을 수행한다. IoT Server 로부터 메시지를 전달받은 Request Broker 는 Message Queue 에 메시지를 저장한 후 Gateway 에 Request 메시지가 있음을 MQTT 를 통해 알려준다. 알림은 받은 Gateway 는 Message Queue 에 있는 Request 메시지를 CoAP 을 통해 확인한 후 IoT Server 에 상태 알림 메시지 (Response 메시지에 대응)를 전달한다.

만약 IoT Server 가 지속적으로 전달할 메시지를 가진 경우 연결 설정을 위한 메시지를 Request Broker 에 전달할 수 있으며, 연결 설정 메시지를 전달받은 Request Broker 는 Message Queue 에 해당 메시지를 저장하지 않고, MQTT 로 바로 Gateway 에 전달한다. 메시지를 수신한 Gateway 는 IoT Server 에 CoAP Observe 메시지를 전달하고, 이후 Broker 를 경유하지 않고 IoT Server 와 통신을 수행한다.

IoT-RMSA 는 일반적인 사물인터넷 플랫폼 구조에 Request Broker 가 추가된 구조로 볼 수 있으며, 이로 인해 기존 사물인터넷 플랫폼에 큰 수정작업 없이 접목할 수 있다. 또한 Request Broker 에 요청 메시지 저장을 위한 Message Queue 를 두어 IoT Server 는 요청메시지 오류 제어에서 자유롭다. 또한 IoT Server 에서 CoAP 메시지를 MQTT 메시지로 변환하기 위한 절차를 수행할 필요가 없으며, CoAP 을 통해 Queue 에 메시지를 저장할 수 있어 요청메시지를 전달하기 위해 TCP 연결설정을 수행하지 않아도 된다. 또한, Request Broker 는 IoT Server 와는 별개로 요청만을 처리하는 노드이므로 필요시에 유연하게 확장하여 부하를 분산함으로써 성능 향상을 도모할 수 있다.

Ⅳ. 제안 기법 검증

본 논문에서 우리는 제안 기법의 검증을 위해 간단한 테스트베드를 구현하였다. 테스트베드는 IoT Server, Broker, Gateway 로 구성되어 있으며, 모두 go-coap 및 paho.mqtt.golang 라이브러리로 구현하였다. 또한, Docker 를 사용하여 실험환경 구축을 수행하였다[3].

그림 3 은 컨테이너 간 교환된 패킷을 캡처한 화면이다. 본 그림에서 IoT Server 는 CoAP 을 통해 Broker 에 요청을 전달하고 있으며, Broker 는 MQTT 로 Gateway 에 해당 사실을 전달한다. MQTT 메시지를 수신한 Gateway 는 IoT Server 에 응답메시지를 CoAP 으로 전달한다. 본 그림을 통해 제안기법의 Information-Flow 가 정상적으로 작동함을 확인할 수 있다.

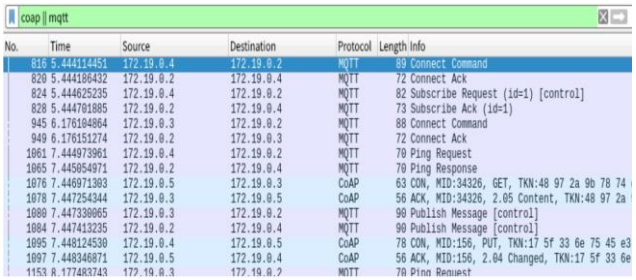

그림 3. 제안 기법을 통한 메시징 패킷 캡처

Ⅴ. 결 론

본 논문에서는 NAT 로 인해 발생하는 통신 문제 해결을 위한 사물인터넷 마이크로 서비스 구조를 설계하고, 간단한 테스트베드를 구현하여 작동 절차를 검증하였다. 본 논문의 제안기법을 통해 실시간 메시징과 함께 부하 분산 모델을 쉽게 적용할 수 있을 것이라 기대한다. 현재 우리는 제안 기법과 LwM2M 간 호환 모듈을 개발하고 있으며, 추후 더욱 다양한 사물인터넷 플랫폼과 호환하기 위한 인터페이스 모듈을 개발할 것이다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW 중심대학사업의 연구결과로 수행되었음(2021-0-01082)

본 연구는 2021 년도 산업통상자원부 및 산업기술평가관리원(KEIT) 연구비 지원에 의한 연구임(20015107)

참 고 문 헌

[1] IETF RFC 7252, “The Constrained Application Protocol (CoAP)”, June, 2014.

[2] ISO/IEC 20922, “Information technology - Message Queuing Telemetry Transport (MQTT) v3.1.1”, June, 2016

[3] lucas-clemente/quic-go: A QUIC implementation in pure go [Internet], https://github.com/lucas-clemente/quic-go.