논문 21-46-11-08 The Journal of Korean Institute of Communications and Information Sciences '21-11 Vol.46 No.11
https://doi.org/10.7840/kics.2021.46.11.1822

NAT 기반 사물인터넷 환경에서 실시간 통신 지원을 위한 MQTT-CoAP 혼합 기법

김 근 수•, 정 중 화*, 고 석 주°

Hybrid Transmission of MQTT and CoAP for Real-Time Communications in NAT-Based IoT Environments

Keun-Soo Kim•, Joong-Hwa Jung*, Seok-Joo Koh°

요 약

NAT(Network Address Translation) 기술을 사용하는 경우 인바운드(inbound, NAT 외부 → NAT 내부) 통신 전에 아웃바운드(outbound, NAT 내부 → NAT 외부) 통신이 먼저 이루어져야 한다. 이때, 일정 시간이 지나서 NAT 매핑 테이블의 세션이 만료되면 NAT 외부 노드는 NAT 내부 노드를 발견할 수 없어서 통신 불가능 상태에 빠지게 된다. 이를 해결하기 위해 LwM2M(Lightweight M2M) 에서는 초기화 과정에서 NAT 내부 노드가 외부 노드에 먼저 요청을 하고 오프라인 상태일 때 연결요청들을 대기열에 저장한 뒤 온라인 상태가 되면 처리하는 방식을 취하고 있다. 하지만 이러한 방식으로는 실시간 서비스 제공이 어려우며, NAT 바인딩 유지를 위한 누적 오버헤드가 크게 발생한다. 이 논문에서는 이러한 문제를 해결하기 위해서 MQTT(Message Queuing Telemetry Transport)와 CoAP(Constrained Application Protocol)를 혼합하여 사용하는 M-CoAP 통신 기법을 제안한다. 테스트베드 실험을 통해 제안 기법을 사용하여 실시간 통신이 가능함을 확인하였고, 기존의 LwM2M 통신 기법과의 비교 분석을 통해 전송 효율이 약 17% 개선될 수 있음을 확인하였다.

Key Words : Internet of Things (IoT), Network Address Translation (NAT), Real-time Communication, Lightweight M2M (LwM2M), Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT)

ABSTRACT

Communication using NAT(Network Address Translation) requires outbound communication (internal node → external node) before performing inbound communication (external node → internal node). After a certain period of time, if the session of the NAT mapping table expires, the external node may not discover the internal node. To address this problem, LwM2M(Lightweight M2M) ensures that an internal node first sends a request to an external node during the initialization process, and that the requests generated in the off-line status are automatically stored in a queue so as to process those requests when they get into the on-line status. However, this approach may not provide real-time communication effectively, and large cumulative overhead may be incurred so as to maintain many NAT bindings for real-time communication. In this paper,

※ 본 연구는 2021년도 한국연구재단 기초연구사업(NRF-2021R1I1A3057509) 지원을 받아 수행됨
◆ First Author : School of Computer Science and Engineering, Kyungpook National University, kks36145919@gmail.com, 학생회원
° Corresponding Author : School of Computer Science and Engineering, Kyungpook National University, sjkoh@knu.ac.kr, 종신회원
* School of Computer Science and Engineering, Kyungpook National University, godopu16@gmail.com
논문번호 : 202105-104-B-RN, Received May 11, 2021; Revised July 28, 2021; Accepted August 10, 2021

to solve this problem, we propose the M-CoAP scheme that combines the two well-known protocols: MQTT(Message Queuing Telemetry Transport) and CoAP(Constrained Application Protocol). From testbed experimentation, we confirmed that the proposed M-CoAP scheme can provide real-time communication. By performance comparison with the existing LwM2M scheme, we also see that the proposed can improve the transmission throughput by 17%.

Ⅰ. 서 론

최근 여러 분야에서 사물인터넷 기술을 융합한 서비스가 활발히 개발되고 있다. Gartner의 보고서에 따르면 전 세계 사물인터넷 장치의 개수는 2018년에 약 39억 개로서 매년 20%씩 증가할 것으로 예상된다[1]. 한편, 사물인터넷 통신을 지원하기 위해 IETF (Internet Engineering Task Force)에서는 CoAP (Constrained Application Protocol) 프로토콜을 제정하였고[2], 이를 기반으로 OMA(Open Mobile Alliance)에서는 센서 네트워크와 M2M(Machine-to-Machine) 환경을 위한 장치 관리 프로토콜인 LwM2M(Lightweight Machine to Machine)을 제정하였다[3].

수많은 사물인터넷 장치를 네트워크 인프라에 연결하려면 각 장치는 식별 가능한 인터넷 주소(IP)를 보유해야 한다. 하지만 IP주소 부족으로 인해 NAT(Network Address Translation) 기술로 사설망을 구성하여 내부 장치에 공인 IP주소가 아닌 사설 IP주소를 할당하는 구조가 일반적이다[4].

하지만 NAT 기술을 이용한 통신에는 몇 가지 제약사항이 있다. 첫 번째로 인바운드 통신(NAT 외부 → NAT 내부)을 수행하기 이전에 아웃바운드 통신(NAT 내부 → NAT 외부)이 먼저 이루어져야 한다. 두 번째로 NAT 내부-외부 노드 간 최초 통신 이후 일정 시간이 지나서 NAT 매핑 테이블의 세션이 만료되면, NAT 외부 노드는 NAT 내부노드를 발견할 수 없어 실시간 통신이 불가능한 상태에 빠지게 된다.

이를 해결하기 위해 LwM2M 기법에서는 초기 연결 과정에서 아웃바운드 수행이 먼저 이루어지게 설계하였다. 또한, 대기열(queue) 상태 기능을 도입하여 NAT 매핑 테이블의 세션이 만료되어도 연결요청을 대기열에 축적하였다가 NAT 내부 클라이언트가 온라인 상태로 변경될 때 대기열의 요청들을 처리하고 있다.

한편, LwM2M 통신은 대기열 대기 송신 방법을 사용하기 때문에 NAT 매핑 테이블의 세션이 만료되면 서버는 클라이언트의 메시지를 수신하기 전까지 통신할 수 없는 문제점이 있다. 따라서, 실시간 통신을 위해서는 세션 만료 방지를 위해 주기적인 통신이 필수적이다. 이때 세션 만료 및 제거를 결정하는 타임-아웃 시간은 TCP 통신 세션보다 UDP 통신 세션이 더 짧기 때문에 통신 유휴 시간이 늘어날수록 LwM2M 통신은 NAT 바인딩을 유지하는 데에 더 큰 누적 오버헤드가 발생한다.

이러한 문제점들은 NAT를 활용하는 사물인터넷 환경에서 실시간 양방향 통신 제공을 어렵게 만든다. 이 논문에서는 NAT 기법을 사용하는 사물인터넷 환경에서 실시간 통신을 제공하기 위한 기법을 제안한다.

특히, 제안 기법에서는 MQTT(Message Queuing Telemetry Transport) 프로토콜과 CoAP 프로토콜을 결합하는 혼합 통신 방식을 사용한다. 제안 기법에서 CoAP 장치의 명령 수신은 MQTT 메시지로, 데이터 전송은 CoAP 메시지로 전송된다. 모든 MQTT 메시지는 MQTT 브로커를 경유하여 교환되고, 브로커는 keep-alive 옵션을 통해 NAT 바인딩 및 연결 상태를 유지한다[5]. 이때 제안 기법은 LwM2M 통신보다 NAT 바인딩을 유지하는 데에 누적 오버헤드가 상대적으로 적은 장점이 있다. 또한, 연결 초기화 이후 NAT 외부의 네트워크 노드가 먼저 NAT 내부의 네트워크 노드로의 통신 시작에 문제가 되지 않으며, 초경량 메시지로 브로커 서버를 중계하여 통신하기 때문에 IoT 서버의 부담을 덜어줄 수 있다. 따라서 NAT 환경의 CoAP 장치에 최적화하여 앞서 언급한 문제점들을 모두 보완할 수 있다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구로써 NAT 기법의 문제점 및 LwM2M 기반의 CoAP 통신 기법을 설명한다. 3절에서는 이 논문에서 제안하는 기법 및 세부 절차들을 기술한다. 4절에서는 테스트베드 실험을 통해 제안 기법이 실시간 통신 기능을 제공할 수 있음을 보여주고, LwM2M 방식과의 성능 비교 분석을 수행한다. 마지막으로 5절에서 결론을 맺는다.

Ⅱ. 관련 연구

2.1 NAT(Network Address Translation)

NAT 기술은 기존 IPv4(Internet Protocol version 4) 주소가 고갈되어감에 따라 이를 해결하는 방안으로 개발되었으며, RFC 1663에서 기본 NAT 유형이 처음 언급되었다[6]. 부족한 공인 IP주소를 여러 개의 사설 IP주소로 변환 및 할당하여 여러 클라이언트가 사용할 수 있도록 한다. IPv4 주소 부족의 또 다른 해결방법인 IPv6 주소는 이미 오래전에 도입되었지만, 인터넷에 배포는 비교적 느려 현재 네트워크 대부분은 여전히 IPv4 주소를 기반으로 이루어져 있다. IPv6 주소를 사용하더라도 사물인터넷 인프라에서의 IPv4 주소 장치와 IPv6 주소 장치 간 통신을 하기에는 어렵다[7].

사물인터넷 장치는 제한된 환경의 저전력, 낮은 품질의 네트워크 등의 상황에서 작동되는 경우가 많으므로 사물인터넷 장치가 직접 IP 전환 메커니즘을 지원하기는 어렵기 때문이다[8]. 따라서 사물인터넷 장치 네트워크에서 IPv4 주소와 IPv6 주소 간 변환 수행의 목적으로도 NAT 기술을 활용한다. 또한, NAT 기술을 사용하면 외부의 불법적 접근 및 내부 정보 공개를 막고 인터넷망에 안전하게 연결할 수 있다. 그러므로 방화벽이나 라우터에 캐리어급 NAT를 활성화하여 사물인터넷 인프라를 사설망으로 구축하는 것이 일반적이다[9]. 현재 기업 내 네트워크망 분리 또는 방화벽으로 인한 불법적 접근 방지 등으로도 널리 사용되고 있으며 가정이나 산업 현장에 상용화되어 있는 공유기도 NAT의 한 종류로서 PAT(Port Address Translation)이다.

이러한 NAT의 타입을 RFC 3489에서 처음으로 제정하였다[10]. RFC 3489에서는 STUN(Simple Traversal of User Datagram Protocol)의 규약과 더불어, NAT 타입을 주소 할당 방식(Mapping)에 따라 크게 Cone NAT와 Symmetric NAT로 총 2가지로 분류하였다.

2.2 사물인터넷 프로토콜

2.2.1 CoAP(Constrained Application Protocol)

CoAP은 IETF에서 제정한 UDP 기반의 사물인터넷 메시지 전송 프로토콜이다[2]. REST(REpresentational State Transfer) 기법을 사용하기 때문에 웹 서비스와 호환성이 좋으며, 제한된 네트워크 통신에 특화되어 적은 전력을 소모하고 신뢰성 있는 통신을 제공한다.

CoAP은 그림 1과 같이 요청(request)과 응답(response)의 상호 전달로 통신한다. CoAP 요청과 해당 요청에 대한 응답은 토큰이라는 필드가 서로 짝을 이룬다. 반면, CoAP 메시지 ID는 각 트랜잭션을 구분한다.

CoAP의 메시지 타입으로는 총 4가지가 있으며(confirmable, non-confirmable, acknowledgement, reset), CoAP Observe 옵션 필드를 이용하여 Polling 방식이 아닌 이벤트 방식 통신이 가능하다[11]. CoAP 서버에 주제(Subject)라는 공간을 할당하여 그 주제에 관심 있는 클라이언트가 주제를 통해 서버로 접근할 수 있다. 클라이언트가 해당 주제로 서버에 Observe 요청을 한다면 그 주제에 관한 내용이 바뀔 때마다 별도의 요청 메시지 없이 서버로부터 변경된 내용을 수신할 수 있다.

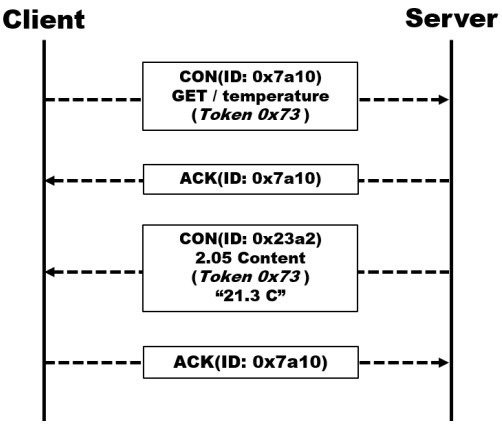

그림 1. CoAP 개요.
Fig. 1. CoAP overview.

2.2.2 MQTT(Message Queuing Telemetry Transport)

MQTT는 발행-구독(Publish-Subscribe) 기반 메시지 큐잉 프로토콜이다[12]. 모바일 기기나 낮은 네트워크 대역폭의 소형 장치들의 연결에 최적화된 메시지전달 프로토콜로서 느리고 품질이 낮은 네트워크에서도 메시지를 안정적으로 전송할 수 있다. TCP 기반의 전송 프로토콜이며 최근 페이스북 메신저 앱에서는 MQTT 방식의 도입을 통해 전송 속도를 대폭 올렸다[13].

먼저 장치에서 MQTT로 통신을 하려면 장치에 MQTT 라이브러리가 설치되어 있어야 하며, 어떤 네트워크로 연결하든지 MQTT 브로커와 연결이 되어 있어야 한다. MQTT 라이브러리는 JavaScript, PHP, C, C++, 안드로이드 등 여러 언어로 개발되어 있다.

본 논문에서의 제안하는 기법에는 JavaScript 언어로 된 MQTT.js 라이브러리를 사용한다. MQTT 브로커 서버는 Mosquitto, HiveMQ, RSMB. VerneMQ 등의 많이 사용되는 브로커 서버들이 있는데 본 논문에서 제안하는 기법에는 제일 경량의 Mosquitto 브로커 서버를 사용한다.

MQTT의 간략한 동작은 그림 2와 같다. Subscriber가 한 토픽에 대해 구독함을 브로커에게 알리면, 브로커는 Publisher가 해당 토픽에 대해 보낸 데이터를 받아서 구독한 Subscriber에게 데이터를 송신해줌으로써 최종적으로 Subscriber는 원하는 데이터를 수신할 수 있고 Publisher는 원하는 데이터를 송신할 수 있다.

또한, MQTT는 토픽을 계층구조로 구성하여 Multicasting 또는 Broadcasting을 할 수 있다[14]. 브로커는 토픽을 / 로 구분하여 하위 계층의 토픽을 추가로 구성할 수 있으며 +를 이용하여 하나의 계층의 모든 토픽을 지칭, #을 이용하여 하위의 모든 계층의 토픽을 지칭할 수 있다.

예를 들어, 그림 3에서 보는 바와 같이 장치 1에서 장치3까지 총 3개의 장치가 있다고 가정하고, 각 장치는 클라이언트라고 하자. 3개의 클라이언트는 차례로 Korea/Device1, Korea/Device2, UnitedStates/Device3을 구독한다. 이때 서버는 Korea/# 이라는 토픽으로 브로커 서버에게 Publish를 하면 / 로 구분된 토픽 중 Korea로 시작되는 모든 토픽을 구독한 장치들에게만 Multicasting을 할 수 있다. 이러한 방식대로 서버는

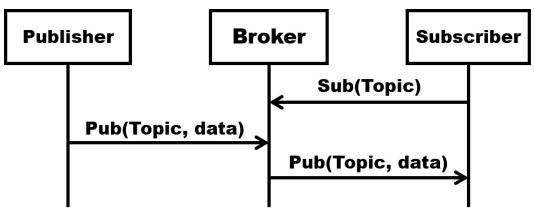

그림 2. MQTT 개요
Fig. 2. MQTT overview.

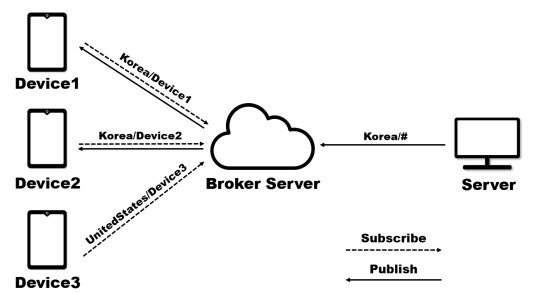

그림 3. MQTT 메시징
Fig. 3. MQTT Messaging.

+, #을 활용하여 해당 토픽을 구독한 클라이언트들에게 브로커 서버를 통해 발행하여 Unicasting, Multicasting, Broadcasting을 필요에 따라 모두 수행할 수 있다.

2.2.3 LwM2M(Lightweight M2M)

LwM2M은 M2M(Machine-to-Machine) 환경의 사물인터넷 장치 관리 및 서비스와 센서 네트워크를 위한 CoAP 기반 OMA(Open Mobile Alliance)의 프로토콜이다[3]. LwM2M은 RESTful 자원 및 데이터 모델을 정의하며 CoAP 기반이기 때문에 DTLS 기반 데이터 전송을 한다. 통신 노드는 LwM2M 서버와 LwM2M 클라이언트로 분류하며 LwM2M 서버는 일반적으로 개인 또는 공용 데이터 센터에 있으며, LwM2M 클라이언트는 사물인터넷 장치에 탑재된다. LwM2M 서버와 LwM2M 클라이언트 간에 논리적 인터페이스 종류로는 총 4가지가 있다.

1) Bootstrap Interface : 부팅 시 LwM2M 서버 정보와 보안키를 부여한다.
2) Registration Interface : LwM2M 클라이언트의 주소 정보 저장 및 액세스 객체를 관리한다.
3) Management Interface : 템플릿(Read, Write, Execute, Discover) 기반 인스턴스 자원을 감시 및 제어한다.
4) Reporting Interface : 리소스 변화 관찰을 위해 Observe 기능과 서버에 신규 데이터 전송을 위해 Notify 기능을 수행한다.

2.3 NAT 방식의 문제점

다양한 NAT 종류로 인한 제어 복잡성과 더불어 NAT 통신을할 시 2가지 문제점이 있다.

첫 번째로 그림 4와 같이 NAT 내부 IoT Node에서 외부 Server로의 통신 시작은 가능하나 외부 Server에

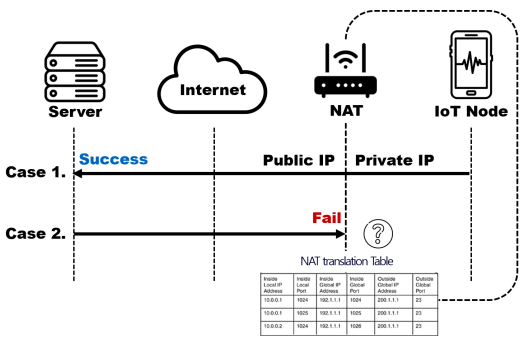

그림 4. 첫 번째 NAT 통신의 문제점
Fig. 4. NAT problem 1.

서 내부 IoT Node로의 통신 시작이 불가능하다. 예를 들어, 공인 IP를 사용하는 서버가 NAT 내부의 사설 IP를 사용하는 클라이언트를 발견할 수 없다.

이를 해결하기 위해 포트 포워딩, ALG(Application Layer Gateway), NAT Traversal(RFC 5128) 기술 등과 LwM2M 프로토콜이 개발되었다.

포트 포워딩 기술은 NAT와 같은 네트워크 게이트웨이를 거칠 때 하나의 IP주소와 포트 번호를 결합해 해당 통신 요청을 전달하는 기술이다. 포트 포워딩 기술을 사물인터넷 분야에 적용해 수많은 장치에 해당하는 포트 번호를 일일이 설정하기에는 어렵다.

ALG 기술은 방화벽 또는 NAT를 강화하는 보안 구 요소로서 특정한 통화량을 분석하여 방화벽을 통과시켜 서비스가 가능하도록 자원을 할당하고 유동적인 방화벽 정책을 설정할 수 있도록 지원하는 기술이다. 현재 대형 사업자에서 사용하는 기술로서 애플리케이션 내용까지 확인해서 매핑 및 필터링을 하는 방법이지만, 잘 알려지지 않은 자체 제작 프로토콜 구현이 불가능하고 PAT 등의 장비 성능에 무리가 가며, 빈번한 패킷 교환을 하는 사물인터넷 환경에 적용하기는 어렵다[15].

NAT Traversal(RFC 5128) 기술에는 STUN (Session Traversal Utilities for NAT), TURN (Traversal Using Relays around NAT), ICE (Interactive Connectivity Establishment) 등이 있다[16].

STUN 기술을 적용한 STUN 서버는 클라이언트의 NAT 또는 방화벽의 유무 및 타입 정보, 공인 IP 정보를 제공함으로써 두 앤드 포인트 간의 P2P 연결을 해준다[17]. 특별한 동작 없이 기존의 다양한 NAT 타입과 함께 작동할 수 있지만, 연결 불가 상황의 대응 방법이 없다. 그러므로 TURN 기술을 활용한 TURN 서버와 같이 사용되는데, P2P 연결이 실패할 경우 TURN 서버는 공인 IP주소를 가지고 NAT 내/외부 간 통신을 지원한다. 두 앤드 포인트 간 패킷 송수신을 중계해주며, 단일 중계 주소로 여러 앤드 포인트와 통신을 할 수 있다[18]. 하지만 모든 패킷이 TURN 서버를 중계하여 전송되기 때문에 지연시간과 대역폭이 짧은 단점이 있다.

ICE 기술은 양 앤드 포인트가 서로 다른 NAT 내부에 존재할 시 연결 불가에 대한 해결방안으로 개발되었다. STUN과 TURN 모두를 사용하며 앤드 포인트의 사설 IP 및 공인 IP와 TURN의 공인 IP, 총 3개의 후보 전송 주소를 기반으로 연결 가능한 쌍을 추적하여 P2P 연결을 해준다[19]. 하지만 ICE 또한 TURN을 사용하기 때문에 TURN의 단점을 그대로 가지고 있으며, 주로 WebRTC(Web Real-Time Communication)를 위해 개발되어 사물인터넷 분야에 적합하지 않다[20]. 또한, 네트워크 장비 회사는 프로토콜 개발 당시에 이러한 NAT Traversal 기술을 추가로 도입하지만, 널리 알려진 프로토콜만 인식 가능하며 최신 프로토콜의 경우이거나 직접 개발한 프로토콜은 인식이 어렵다.

이 외에 NAT Traversal 방법들이 많이 존재하지만, 사물인터넷 통신에 적합한 NAT Traversal 방법이 없다. 사물인터넷에 적합하게 개발된 LwM2M 프로토콜은 이러한 NAT Traversal 방법과 문제점을 일부 해결하기 위해 초기화 과정에서 NAT 내부 클라이언트가 NAT 외부 서버에게 등록 메시지를 전송하여 아웃-바운드 통신이 먼저 수행되도록 설계되었다.

두 번째로 NAT 내부에서 외부로 통신이 선 수행되어 NAT 바인딩이 되어도, 이후 통신 유휴 시간이 길어지면 NAT 바인딩이 끊어져 NAT 외부에서 내부로 통신이 불가하다. NAT 장비는 제조사마다 NAT 매핑 테이블의 바인딩 타임아웃 설정값이 다르며, 양측의 네트워크 노드들이 잠시 쉬고 있어서 통신이 초기 설정된 타임아웃의 시간 동안 발생하지 않으면 그림 5와 같이 NAT 장비는 통신이 종료됐다고 판단하여 세션 정보를 제거한다. 따라서 NAT 외부의 네트워크 노드는 NAT 내부의 네트워크 노드를 탐색할 수 없어서 통신이 불가하다.

이를 해결하기 위해 LwM2M 프로토콜은 대기열 상태 기능을 도입하였다. LwM2M 서버와 NAT 내부의 LwM2M 클라이언트가 오랜 시간 통신을 하지 않을 시, NAT 세션 정보가 삭제되고 서버는 대기열 상태에 돌입한다. 서버는 이후 요청을 즉시 처리하지 않고 클라이언트가 온라인이 될 때까지 대기하며 요청들을 대기열에 저장한다. 클라이언트가 업데이트 메시

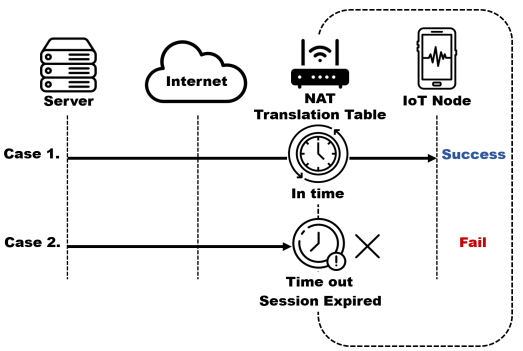

그림 5. 두 번째 NAT 통신의 문제점
Fig. 5. NAT problem 2.

지를 서버에 송신하여 클라이언트가 온라인 상태가 되면 서버는 대기 중인 요청들을 순서대로 처리한다.

하지만, 이러한 LwM2M 통신은 대기열 대기 송신 방법을 사용하기 때문에 서버는 클라이언트의 업데이트 메시지를 수신하기 전까지 클라이언트와 실시간 통신을 할 수 없는 문제점이 있다. 세션 만료 방지를 위해 주기적인 통신을 하더라도 CoAP 통신을 하는 LwM2M은 비연결형 통신이기 때문에 통신 유휴 시간이 길어질수록 큰 누적 오버헤드가 발생한다. 또한, 네트워크 장비 회사마다 NAT 매핑 테이블의 초기 세션 만료 시간 설정값은 상이하며 세션 만료 시간을 전송 프로토콜을 기준으로 결정하는데, 비연결형 기반 통신인 UDP 통신보다 연결형 기반 통신인 TCP 통신의 만료 타이머가 대체로 더 길다. 이러한 설정값을 수많은 사물인터넷 장치와 연결된 네트워크 장비를 서비스마다 일일이 설정하기에는 어렵다.

따라서 CoAP 통신을 하는 LwM2M은 NAT 바인딩을 유지하더라도 상대적으로 빈번한 메시지 교환 때문에 통신에 비효율적인 문제가 있다. 그뿐만 아니라, 주기적인 요청으로 인해 서버의 처리량이 커지고 클라이언트의 수가 증가한다면 해당 정보 값을 모두 저장해야 하고 그에 따른 명령 또한 대기열에 저장해야 하므로 메모리의 사용량에 크게 영향을 끼친다.

Ⅲ. 제안하는 M-CoAP 기법

3.1 M-CoAP 통신 구조

본 논문에서는 제한된 환경에서 NAT 내부 사물인터넷 장치들과 실시간 양방향 통신 지원 및 게이트웨이의 누적 오버헤드 감소를 위해 LwM2M 통신의 문제점을 개선하여 CoAP과 MQTT를 결합한 M-CoAP 통신 기법을 제안한다. 제안 기법에는 일반적인 사물인터넷 장치들의 작동 환경을 고려하여 데이터를 수집 및 관리하는 게이트웨이를 통한 통신 구조로 구성하였다[21].

그림 6은 제안하는 M-CoAP 통신 기법을 설명하고 있다. 제안 기법에는 NAT 바인딩 유지에 사용되는 게이트웨이의 누적 오버헤드를 줄이고, NAT 외부에서 내부로 실시간 명령 처리를 하기 위해 MQTT 브로커를 추가하였다. 서버는 제어 명령 및 데이터 요청을 MQTT 메시지로 송신하고, 해당 응답을 CoAP 메시지로 수신한다. 다음은 각 노드의 역할과 상세 설명이다.

- *IoT Server* : 사용자의 명령을 처리하고 관리하는 모든 IoT Node들의 정보 및 상태를 보관하는 노드이다. MQTT 브로커는 사용자의 명령 처리 및 사물인터넷 장치들의 상태와 정보를 보관하는 노드이다. 게이트웨이의 CoAP 메시지를 수신할 수 있도록 CoAP 서버가 내부적으로 구성되어 있고 IoT Node에 명령을 전송할 수 있도록 MQTT 브로커와 연결을 맺는다.
- *MQTT Broker* : IoT Server의 명령을 수신 및 중계하여 NAT의 공인 주소 정보로 전달하는 노드이다. 명령 정보를 MQTT 메시지로 수신하여 해당 목적지와 일치하는 M-CoAP 게이트웨이로 MQTT 메시지로 송신한다. 또한, IoT Server와 M-CoAP 게이트웨이 간 NAT 바인딩을 유지하는 역할을 한다.
- *NAT* : 하나의 공인 IPv4 주소를 여러 사설 IPv4 주소들로 변환하여 내부 사설망의 노드들을 인터넷망에 연결해주는 노드이다. M-CoAP 게이트웨이와 IoT Node를 내부망으로 연결하고 외부망인 인터넷망과 중계해준다.
- *M-CoAP Gateway* : NAT 내부의 사설망에 위치하며 해당 네트워크망에 연결된 모든 사물인터넷 장치들을 관리한다. 실시간으로 사물인터넷 장치들의 데이터를 수집하고 서버에게 수신받은 제어 명령을 실시간으로 처리한다.
- *IoT Node* : 실시간 데이터 수집을 하는 CoAP 기반 통신 기기로써 여러 종류의 사물인터넷 장치가 될 수 있다. 장치에 연결된 센서로 수집한 데이터들을 M-CoAP 게이트웨이에 실시간으로 전송한다. 또한, M-CoAP 게이트웨이로부터 수신한 명령을 실시간으로 처리한다.

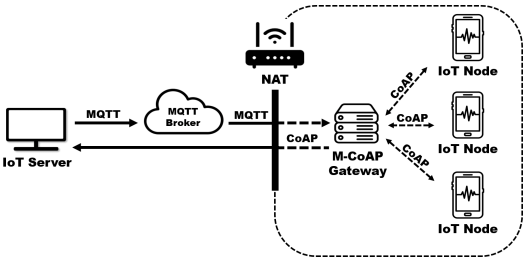

그림 6. M-CoAP 네트워크 모델
Fig. 6. M-CoAP network model.

3.2 M-CoAP 통신 절차

3.2.1 초기화 과정

그림 7은 NAT 내부노드들의 통신 초기화 과정으로, M-CoAP 게이트웨이의 IoT Node 탐색과 양방향

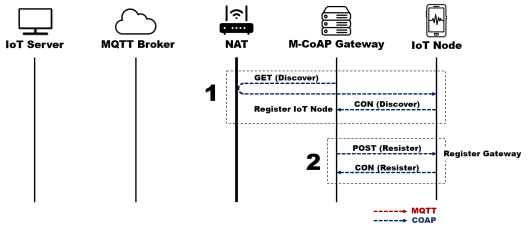

그림 7. M-CoAP 장치 초기화
Fig. 7. Device initialization in M-CoAP.

등록과정의 데이터 흐름이다.

먼저 박스1은 게이트웨이의 IoT Node 탐색 과정이다. IoT Node는 CoAP 메시지를 수신할 수 있도록 서버를 열고 대기한다. 게이트웨이는 사설망에 GET CoAP 메시지를 브로드 케스트로 송신한다. 이를 수신한 IoT Node는 CON CoAP 메시지에 자신의 주소와 정보를 담아 M-CoAP 게이트웨이에 응답한다. 게이트웨이는 수신한 IoT Node의 정보를 등록한다.

박스2는 IoT Node의 게이트웨이 등록과정이다. 게이트웨이는 등록한 모든 장치에 자신의 주소 정보를 POST CoAP 메시지로 송신한다. IoT Node는 수신한 주소 정보로 게이트웨이를 등록하고 CON CoAP 메시지로 응답하여 완료됨을 알린다.

그림 8은 NAT 내/외부 노드 간 통신 초기화 과정으로, MQTT 브로커를 통해 M-CoAP 게이트웨이와 서버 간 양방향 연결 및 등록과정의 데이터 흐름이다.

박스3은 서버와 게이트웨이가 MQTT 브로커에 연결을 맺고 게이트웨이가 구독하는 과정이다. 게이트웨이의 구독 토픽은 해당 게이트웨이의 위치 및 정보를 나타낸다. 게이트웨이의 구독 토픽 구조는 표 1과 같다. 게이트웨이의 구독 토픽 SUBTOPIC의 구조는 '국가 코드/서비스 코드/해당 게이트웨이 고윳값'이다. 국가 코드는 ISO 3166-1 Alpha-3을 따르며 각 국가를 나타내는 알파벳 세 글자이다. 서비스 코드는 각 행정시, 구로 나뉘어 있는 지역 고유의 세 글자 10진

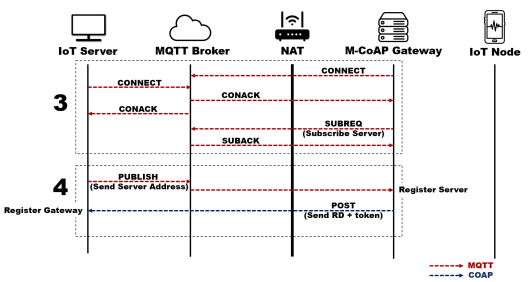

그림 8. IoT Server와 M-CoAP Gateway 초기화
Fig. 8. Initialization of IoT Server/M-CoAP Gateway.

표 1. MQTT 메시지 및 토픽 구조
Table 1. Format of MQTT messages and topics.

게이트웨이 구독 토픽 구조	국가 코드/서비스 코드/게이트웨이 고윳값
토픽 예시 1)	USA/105/
토픽 예시 2)	KOR/002/b213ee6e-e68d-435f-b0b2-fb30a720f05d
서버 발행 메시지 구조	해당 명령어#세부 명령어
메시지 예시 1)	READ#SEONSOR1
메시지 예시 2)	CONTROL#TEMPUP

수 코드이다. 게이트웨이 고윳값은 게이트웨이마다 존재하는 고윳값을 나타내며 임의의 UUID(Universally Uniques Identifier)값을 가진다.

박스4는 초기 연결 설정이 끝난 후, 서버와 게이트웨이의 양방향 등록과정이다. 서버는 CoAP 서버를 열고 제어하고자 하는 지역의 정보를 SUBTOPIC 구조로 토픽 설정을 한다. 설정한 토픽으로 PUBLISH MQTT 메시지에 서버의 주소 정보를 담아 발행한다. 메시지의 형식은 'REGISTER#서버 주소'이다. 이때 사용하는 발행 메시지의 구조는 표 1과 같이 '해당 명령어#세부 명령어' 이다. MQTT 브로커는 이를 수신하고 해당 정보를 구독한 게이트웨이에 중계 전송한다. 게이트웨이는 수신한 메시지에 담긴 주소 정보로 서버를 등록하고, 해당 주소로 접속해 자신이 관리하는 IoT Node들의 정보와 구독 정보인 SUBTOPIC을 POST CoAP 메시지로 전송한다. 이를 수신한 서버는 게이트웨이의 구독 정보와 해당 게이트웨이가 관리하는 IoT Node들의 정보를 등록하고 CON CoAP 메시지로 완료했음을 알리며 초기화 과정을 완료한다.

3.2.2 데이터 수집 과정

그림 9는 서버가 IoT Node의 데이터를 실시간으로 수집하는 과정의 데이터 흐름이다.

서버는 데이터를 수집하고자 하는 IoT Node가 연결된 게이트웨이의 정보를 찾는다. '국가 코드/서비스 코드/해당 게이트웨이 고윳값'을 토픽으로 설정한다. 설정한 토픽으로 PUBLISH MQTT 메시지에 수집 명령을 담아 발행한다. 메시지의 형식은 'READ#해당 IoT Node 정보' 이다. MQTT 브로커는 이를 수신하고 해당 게이트웨이에 중계 전송한다. 게이트웨이는 받은 명령에 해당하는 IoT Node에 Observe 옵션을 추가하여 GET 메시지를 전송한다. IoT Node는 센서로부터 수집한 데이터의 변화가 있을 때마다 CON 메시지에 데이터를 담아 게이트웨이에 전송한다. 게이트

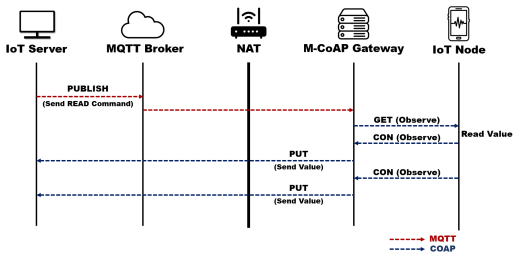

그림 9. IoT Server의 IoT Node 데이터 수집
Fig. 9. Data collection of IoT Server.

웨이는 IoT Node로부터 CON 메시지를 수신할 때마다 서버에게 PUT 메시지를 실시간으로 전송한다.

3.2.3 장치 제어 과정

그림 10은 서버가 실시간으로 IoT Node를 제어하는 과정의 데이터 흐름이다.

서버는 제어하고자 하는 IoT Node가 연결된 게이트웨이의 정보를 찾는다. '국가 코드/서비스 코드/해당 게이트웨이 고윳값'을 토픽으로 설정한다. 설정한 토픽으로 PUBLISH MQTT 메시지에 제어 명령을 담아 발행한다. 메시지 형식은 'CONTROL#해당 IoT Node#수행할 ACTION'이다. MQTT 브로커는 이를 수신하고 해당 게이트웨이에 중계 전송한다. 게이트웨이는 받은 명령에 해당하는 IoT Node에 PUT CoAP 메시지로 제어 명령을 송신한다. IoT Node는 수신한 제어 명령을 처리하고 현재 상태 값을 CON CoAP 메시지로 응답한다. 이를 수신한 게이트웨이는 서버에 PUT CoAP 메시지로 IoT Node의 상태를 업데이트하고 완료 응답으로 CON CoAP 메시지를 수신한다.

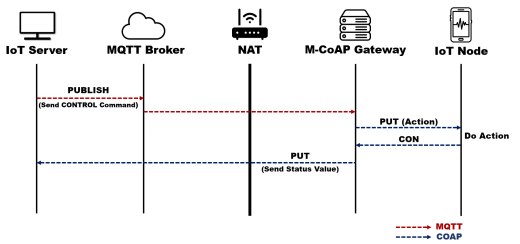

그림 10. IoT Server의 IoT Node 제어
Fig. 10. IoT Server's control of IoT Node.

3.2.4 NAT 바인딩 유지 과정

그림 11은 MQTT 브로커와 M-CoAP 게이트웨이 간 NAT 바인딩 유지 과정의 데이터 흐름이다.

M-CoAP 게이트웨이는 NAT 매핑 테이블의 세션이 만료되기 전에 MQTT 브로커에 페이로드가 비어

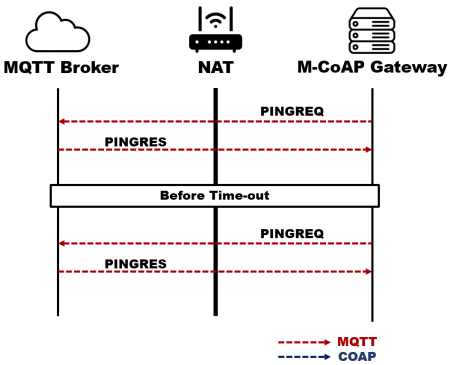

그림 11. MQTT 브로커와 M-CoAP 게이트웨이 간 NAT 바인딩 유지
Fig. 11. Maintenance of NAT binding between MQTT broker and M-CoAP gateway.

있는 PINGREQ 및 PINGRES 메시지를 주기적으로 송신하여 NAT의 타임아웃 세션 갱신 및 NAT 바인딩을 유지한다.

NAT 매핑 테이블은 세션 만료 시간을 주로 전송 프로토콜을 기준으로 결정한다. 비연결형 기반 통신인 UDP보다 연결형 기반 통신인 TCP의 만료 시간을 더 길게 설정하기 때문에 UDP 기반인 CoAP 통신보다 TCP 기반인 MQTT 통신의 만료 시간이 더 길다. 따라서, 본 논문에서 제안하는 기법은 세션을 갱신하기 위해 연결형 기반 통신인 TCP 기반의 MQTT 메시지로 구성하였다.

Ⅳ. 테스트베드 실험 및 성능 분석

4.1 실험용 테스트베드 구성

이 절에서는 기존 NAT 환경의 LwM2M 통신 기법과 본 논문에서 제안하는 M-CoAP 통신 기법의 실험 환경을 설명한다. 기존 기법과 본 논문에서 제안하는 기법의 구조는 각각 그림 12, 13과 같다. NAT 내부의 모든 노드는 802.11n 무선랜(WiFi)으로 연결 및 구현

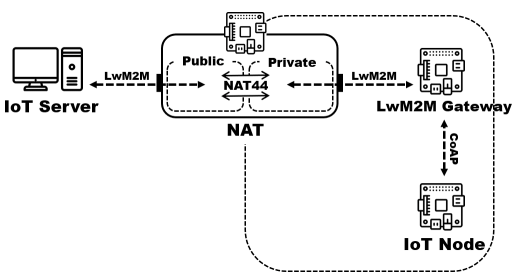

그림 12. 기존 LwM2M 기법의 테스트베드 구조
Fig. 12. Testbed for the existing LwM2M.

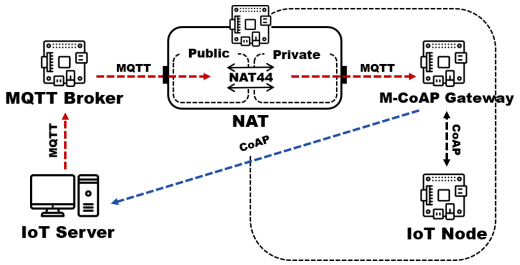

그림 13. 제안하는 M-CoAP 기법의 테스트베드 구조
Fig. 13. Testbed for proposed M-CoAP.

하여 사설망으로 구축하였다. 본 논문에서는 공인 IPv4 주소와 사설 IPv4 간의 주소 변환을 'NAT 44'로 명칭한다.

표 2는 제안 기법을 검증하기 위한 테스트베드 노드 구현 환경이다. 실제 사물인터넷 장치를 고려하여 상대적으로 저장 공간이 적고 구버전의 라즈베리파이 보드를 사용하였다. 또한, NAT 기능을 구현하기 위해 WiFi AP 기능을 제공하는 hostapd 패키지와 DHCP/DNS 기능을 위해 dnsmasq 패키지를 사용하였다. 프로토콜 실험 환경은 Javascript 기반으로 구성하였기 때문에 mqtt, lwm2m-node-lib, coap 패키지를 설치하여 사용하였다. MQTT 브로커는 C언어 기반으로 개발된 Mosquitto로 구현하였으며 실험을 위해 공인망을 192.168.50.0/24 네트워크로 가정하고 NAT를 통해 변환된 사설망을 172.16.0.0/24 네트워크로 구축하였다.

표 2. 테스트베드 구현 환경
Table 2. Testbed implementation environment

	IoT Server	MQTT Broker	NAT	M-CoAP/ LwM2m Gateway	IoT Node
OS	Windows 10	Raspbian 10 buster	Raspbian 10 buster	Raspbian 10 buster	Raspbian 10 buster
Processor	Intel(R) CoreTM i7-8700	Raspberry Pi 4 Model B	Raspberry Pi 4 Model B	Raspberry Pi 4 Model B	Raspberry Pi 3 Model B+
RAM	24GB	8GB	8GB	8GB	4GB
nodejs	14.16.0v	10.23.1v	-	10.23.1v	10.23.1v
mosquitto	-	1.6.10v	-	-	-
mqtt	4.2.6v	4.2.6v	-	4.2.6v	-
lwm2m-node-lib	1.3.0v	-	-	1.3.0v	-
coap	0.24.0v	-	-	0.24.0v	0.24.0v
IP	192.168.50.110	192.168.50.130	192.168.50.120/172.16.0.1(AP)	172.16.0.129	172.16.0.130

4.2 제안 기법의 기능 검증 및 성능 비교 분석

이 절에서는 성능 척도로서 기존 기법과 제안 기법의 통신 유휴 시간을 길게 하여 NAT 외부에서 NAT 내부로의 실시간 통신 가능 유무를 확인하였다. 또한, NAT 바인딩 유지를 위한 누적 데이터양과 RTT(Round-Trip Time)를 측정하였다.

그림 14는 기존 기법을 사용할 때 첫 연결 수립 이후 일정 시간이 지난 뒤 패킷 교환 실패를 NAT에서 Wireshark로 캡처한 결과이다.

NAT는 첫 통신을 위한 요청 패킷을 수신하면 1, 2번째 패킷과 같이 Source IP를 자신의 이더넷 IP주소로 변환하여 전송한다. IP주소 172.16.0.129를 가진 NAT 내부의 게이트웨이가 IP주소 192.168.50.110을 가진 NAT 외부의 IoT Server에 CoAP POST 메시지를 송신하여 등록 신청을 한다. NAT는 이를 수신하여 Source IP를 자신의 이더넷 IP주소인 192.168.50.120으로 변하여 IoT Server에 송신한다.

빨간 박스 부분은 첫 연결 수립 시간인 14:06:50에서 40초가 지난 뒤의 통신 부분이다. NAT 외부의 IoT Server가 NAT 내부의 게이트웨이에 CoAP GET 메시지로 요청을 보내지만, ICMP 메시지로 'Destination unreachable (Port unreachable)' 의 이유로 모든 패킷이 누락됨을 확인할 수 있다. 리눅스 커널 내부 netfilter 프레임워크에 UDP 타임아웃 주기가 30초이기 때문에 유효 통신 없이 30초가 지나면 NAT 매핑 테이블의 해당 정보가 삭제되기 때문이다.

그림 15는 제안 기법을 사용할 때 첫 연결 수립 이후 일정 시간이 지난 뒤 패킷 교환 성공을 NAT에서 Wireshark로 캡처한 결과이다.

기존 기법의 통신과 같이 NAT는 Source IP를 자

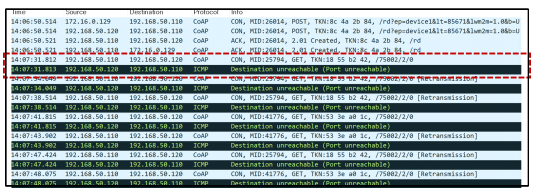

그림 14. 기존 기법의 통신 불가 확인
Fig. 14. Inability to communicate of existing scheme.

그림 15. 제안 기법의 실시간 통신 확인
Fig. 15. Communication of the proposed scheme.

신의 이더넷 IP주소로 변환하여 전송한다. 첫 통신 시작을 위해 IP주소 172.16.0.129를 가진 NAT 내부의 게이트웨이가 IP주소 192.168.50.120을 가진 NAT 외부의 MQTT 브로커에 연결 신청을 하고 ACK(Acknowledgement) 패킷을 통해 MQTT 브로커와 연결이 성공했음을 확인할 수 있다. 이후 게이트웨이는 구독 요청 및 응답 패킷을 수신하며, 국가 코드 및 서비스지역과 UUID가 결합한 토픽을 구독하도록 구현하였다. 본 실험에서는 예시로 “KOR/003/30c692e3-861f-11eb-9a95-f8e4e37c215c”를 구독하였고, IoT Server가 해당 토픽으로 발행하여 게이트웨이를 제어한다.

파란 박스 부분은 첫 연결 수립 시간인 14:09:55에서 59초가 지난 뒤의 통신 부분이다. NAT 외부의 IoT Server는 MQTT 브로커를 중계하여 NAT 내부의 게이트웨이로 MQTT 메시지에 명령을 담아 송신한다. 이를 게이트웨이가 수신함을 확인할 수 있으며 NAT 매핑이 성공적으로 작동하여 실시간 통신을 할 수 있음을 확인하였다. 이후 게이트웨이는 수신한 명령을 처리하여 CoAP 메시지에 결과 메시지를 담긴 JSON 파일을 IoT Server에 송신하며, 최종적으로 통신이 성공적으로 완료됨을 확인할 수 있다. 리눅스 커널 내부 netfilter 프레임워크에 TCP 타임아웃이 432,000초이기 때문에 기존 기법보다 해당 정보가 NAT 매핑 테이블에 오래 남기 때문이다.

그림 16은 기존 기법과 제안 기법의 NAT 바인딩 유지를 위해 발생한 시간당 누적 데이터양을 나타낸다.

본 실험에서는 두 기법 모두 30초 지점부터 통신을 시작하였고 이후 데이터 요청이 발생하지 않고 NAT 바인딩만을 유지하기 위해 발생하는 누적 데이터양을 측정하였다. TCP 기반의 제안 기법이 UDP 기반의 기존 기법보다 첫 연결 수립에 발생하는 데이터가 많음을 확인할 수 있다. IoT Server와 Gateway 양측 모두가 MQTT 브로커와 연결을 수립해야 하므로 30초 지점에 기존 기법(142 bytes)보다 제안 기법(460 bytes)은 약 3.2배 많은 데이터가 발생함을 확인할 수 있었다.

하지만, NAT 매핑 테이블의 정보 값을 갱신하기 위해 30초에 123 bytes 데이터가 1회 교환되는 기존 기법에 반해, 60초에 206 bytes의 데이터가 1회 교환되는 제안 기법이 시간이 지날수록 누적 데이터양이 적어짐을 확인하였다. 약 540초 지점에 제안 기법(2108 bytes)이 기존 기법(2233 bytes)보다 누적 데이터양이 더 적어짐을 확인하였다. 이후 유효 통신 휴식 시간이 길어질수록 누적 데이터양의 차이는 더 증가함을 확인하였다. 8시간 동안 두 앤드 포인트가 유효 통신을 하지 않을 시 NAT 바인딩 유지를 위한 누적 데이터는 약 20,000 bytes의 차이가 발생하며, 제안 기법의 누적 데이터양이 기존 기법보다 약 17% 적음을 확인하였다.

그림 17은 기존 기법과 제안 기법의 패킷 손실률에 따른 누적 데이터양을 비교한 결과이다.

그림에서 600초 동안 NAT 바인딩 유지를 위해 교환된 데이터양에 대한 비교결과를 보여준다. 바인딩 유지를 위해 기존 기법은 30초마다 통신을 하고, 제안 기법은 60초마다 통신을 수행하기 때문에 전제적으로 기존 기법보다 제안 기법에서 교환되는 데이터의 양

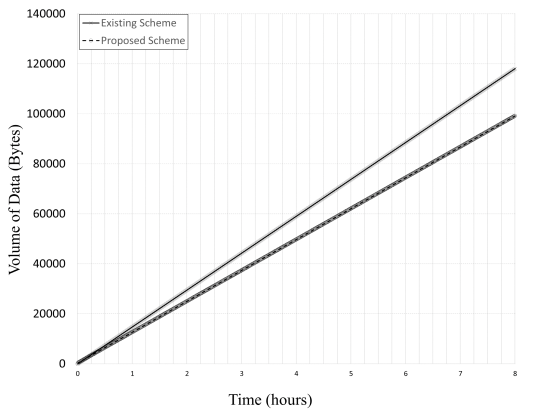

그림 16. 기존 기법과 제안 기법 비교: NAT 바인딩 유지를 위한 시간당 누적 데이터양
Fig. 16. Comparison of the existing and proposed schemes: Accumulated volume of data per hour for maintaining NAT bindings.

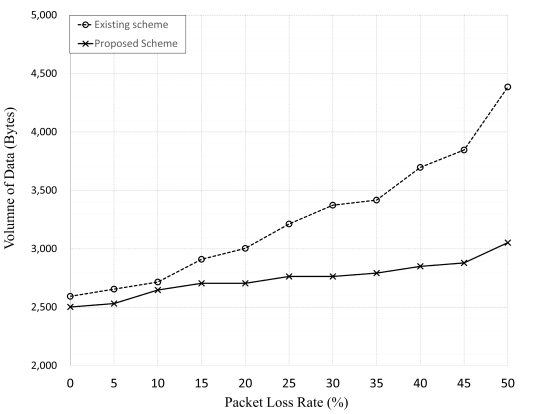

그림 17. 기존 기법과 제안 기법 비교: 패킷 손실률에 따른 누적 데이터양
Fig. 17. Comparison of the existing and proposed schemes: Accumulated volume of data for different packet loss rates.

이 적음을 확인하였다. 특히, 기존 기법의 경우 빈번하게 통신해야 하므로 패킷 손실률이 높을수록 재전송 패킷을 빈번하게 보내게 되고 이로 인해 교환되는 누적 데이터양이 증가하는 경향이 있다. 그림에서 또한 패킷 손실률이 증가할수록 제안 기법과 기존 기법과의 성능 차이가 커짐을 알 수 있다.

V. 결 론

본 논문에서는 NAT 환경에서 LwM2M 통신의 문제점을 분석하고 MQTT와 CoAP 프로토콜을 혼합 적용하는 기법을 제안하였다.

기존의 LwM2M을 활용한 통신 기법은 통신 유휴 시간이 길어져 NAT 바인딩이 끊어졌을 때를 대비하여 Queue Mode가 존재한다. 하지만 이 기법을 사용하는 경우 NAT 내부의 노드에서 NAT 외부로 송신된 패킷을 기다리며 무제한 대기 상태에 돌입되기 때문에 실시간 통신이 불가하다. 이를 해결하기 위해 NAT 바인딩을 유지하더라도 NAT 매핑 테이블 정보가 짧은 시간 내에 삭제되기 때문에 주기적인 갱신을 위해 많은 오버헤드가 소요된다.

한편, 본 논문에서 제안하는 MQTT와 CoAP을 결합한 M-CoAP 하이브리드 기법을 사용하면 상대적으로 긴 시간 동안 NAT 매핑 테이블에 정보가 존재하며, NAT 바인딩을 위한 두 종단 노드간에 주기적인 갱신을 빈번하게 할 필요가 없다. 따라서 기존 기법보다 NAT 바인딩 유지를 위해 발생하는 누적 데이터양이 적고 기존 기법과 차이는 시간에 비례해 커짐을 확인하였다. 또한, 기존 REST 구조로 구현되어 있는 CoAP을 통해 저사양 장치에 그대로 활용할 수 있으며, TCP 기반 통신을 최소로 사용하는 장점이 있다. 이로써 장기간 통신 유휴 상태의 사물인터넷 장치에 실시간 제어가 필요한 환경에는 제안 기법을 통한 통신이 큰 이점을 가지고 있음을 확인하였다.

향후 연구로서 본 논문에서 제안한 리바인딩 기법을 보다 다양한 환경에서 확장 적용하고 이에 대한 성능을 비교 분석하는 연구를 수행할 필요가 있다.

References

[1] *Analysts to Explore How IoT Will Accelerate Digital Transformation Initiatives*, Gartner IT Symposium/Xpo (2019), Retrieved Apr., 30, 2021 from https://www.gartner.com

[2] IETF RFC 7252, "*The Constrained Application Protocol (CoAP)*," 2014.

[3] Open Mobile Alliance, "*LwM2M v1.1 - Lightweight Machine to Machine*," (2019), Retrieved Aug., 30, 2021 from http://www.openmobilealliance.org/release/LightweightM2M/Lightweight_Machine_to_Machine-v1_1-OMASpecworks.pdf

[4] *The networked industrial enterprise*, Ericsson Mobility Report (2020), Retrieved Apr., 30, 2021 from https://www.ericsson.com/en/mobility-report/reports

[5] Wikipedia, *Keepalive*, Retrieved Apr., 30, 2021 from https://ko.wikipedia.org/wiki/keepalive

[6] L. Zhang, *A Retrospective View of NAT* (2007), Retrieved Apr. 30, 2021 from https://ko.wikipedia.org/wiki/keepalive

[7] Oscar Novo, "*Making Constrained Things Reachable: A Secure IP-Agnostic NAT Traversal Approach for IoT*," 2018.

[8] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Busin. Horizons*, vol. 58, no. 4, pp. 431-440, 2015.

[9] I. Choi, "IPsec support for NAT-PT in IPv6 transition mechanisms," *J. KICS*, vol. 30, no. 11B, pp. 736-743, 2005.

[10] IETF RFC 3489, "*STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)*," 2003.

[11] IETF RFC 7641, "*Observing Resources in the Constrained Application Protocol (CoAP)*," 2015.

[12] OASIS Standard, "*MQTT Version 5.0*," (2019), Retrieved Aug., 30, 2021 from https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[13] Facebook, "*Building Facebook Messenger*," (2011), Retrieved Aug., 30, 2021 from https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920

[14] HiveMQ, "*MQTT Topics & Best Practices - MQTT Essentials: Part 5*," (2019), Retrieved Aug., 30, 2021 from https://www.hivemq.com/

blog/mqtt-essentials-part-5-mqtt-topics-best-practices/

[15] N. Verma M. kashyap, and A Jha, "Extending port forwarding concept to IoT," *2018 ICACCCN IEEE*, Greater Noida, India, Oct. 2018.

[16] IETF RFC 5128, "*State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)*," 2008.

[17] IETF RFC 8489, "*Session Traversal Utilities for NAT (STUN)*," 2020.

[18] IETF RFC 5766, "*Traversal Using Relays around NAT (TURN):Relay Extensions to Session Traversal Utilities for NAT (STUN)*," 2010.

[19] IETF RFC 5245, "*Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols*," 2010.

[20] Hans-Christer Holmberg, "*NAT Traversal for Constrained IoT*," 2019.

[21] H. Chen, X. Jia, and H. Li, "A brief introduction to IoT gateway," *IET ICCTA*, Beijing, Oct. 2011.

김 근 수 (Keun-Soo Kim)

2020년 2월 : 경북대학교 컴퓨터학부 공학사

2020년 3월~현재 : 경북대학교 컴퓨터학부 석사과정

<관심분야> IoT, 통신프로토콜, 네트워크 보안

[ORCID:0000-0001-6832-005X]

정 중 화 (Joong-Hwa Jung)

2016년 2월 : 경북대학교 컴퓨터학부 공학사

2018년 2월 : 경북대학교 컴퓨터학부 공학석사

2018년 3월~현재 : 경북대학교 컴퓨터학부 박사과정

<관심분야> IoT, QUIC, 오픈소스 SW

[ORCID:0000-0002-5828-9551]

고 석 주 (Seok-Joo Koh)

1992년 2월 : KAIST 경영과학과 공학사

1994년 2월 : KAIST 경영과학과 공학석사

1998년 8월~2004년 2월 : ETRI 표준연구센터 선임연구원

2004년 3월~현재 : 경북대학교 컴퓨터학부 교수

<관심분야> IoT, QUIC, SCTP, 국제표준화

[ORCID:0000-0003-3429-2040]