Constrained Application Protocol (3)

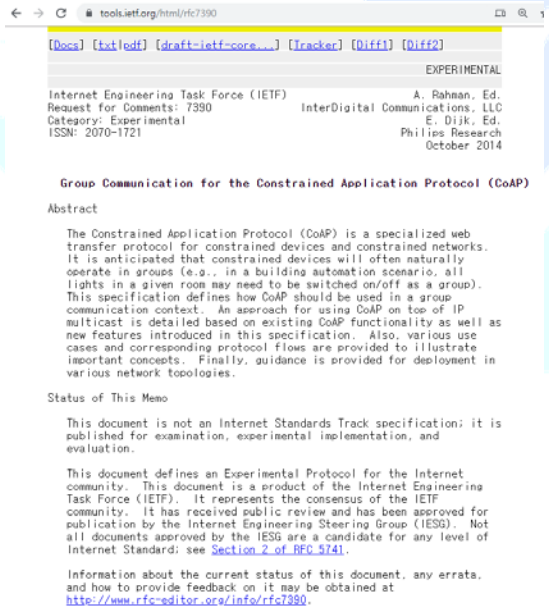
Kim, Eui-Jik

Contents

- Introduction
- Group Communication
- Block-wise Transfer

Introduction

- **IETF CoRE 워킹 그룹은 CoAP외에 여러 가지 프로토콜을 추가로 개발**
  - Group Communication for the CoAP (RFC 7390)
    - 다수의 오브젝트 제어
    - https://tools.ietf.org/html/rfc7390
  - Block-wise transfers in CoAP (RFC 7959)
    - 큰 사이즈 데이터 전송
    - https://tools.ietf.org/html/rfc7959



[Docs] [txt|pdf] [draft-ietf-core...] [Tracker] [Diff1] [Diff2]

EXPERIMENTAL

Internet Engineering Task Force (IETF) — A. Rahman, Ed.
Request for Comments: 7390 — InterDigital Communications, LLC
Category: Experimental — E. Dijk, Ed.
ISSN: 2070-1721 — Philips Research
October 2014

Group Communication for the Constrained Application Protocol (CoAP)

Abstract

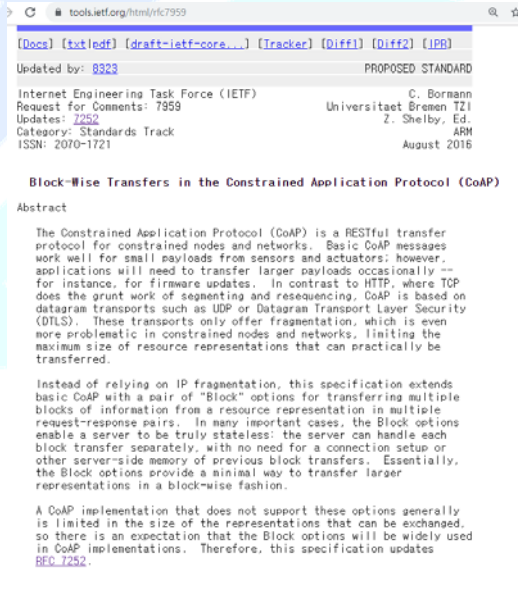
The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for constrained devices and constrained networks. It is anticipated that constrained devices will often naturally operate in groups (e.g., in a building automation scenario, all lights in a given room may need to be switched on/off as a group). This specification defines how CoAP should be used in a group communication context. An approach for using CoAP on top of IP multicast is detailed based on existing CoAP functionality as well as new features introduced in this specification. Also, various use cases and corresponding protocol flows are provided to illustrate important concepts. Finally, guidance is provided for deployment in various network topologies.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7390.



[Docs] [txt|pdf] [draft-ietf-core...] [Tracker] [Diff1] [Diff2] [IPR]

Updated by: 8323 — PROPOSED STANDARD

Internet Engineering Task Force (IETF) — C. Bormann
Request for Comments: 7959 — Universitaet Bremen TZI
Updates: 7252 — Z. Shelby, Ed.
Category: Standards Track — ARM
ISSN: 2070-1721 — August 2016

Block-Wise Transfers in the Constrained Application Protocol (CoAP)

Abstract

The Constrained Application Protocol (CoAP) is a RESTful transfer protocol for constrained nodes and networks. Basic CoAP messages work well for small payloads from sensors and actuators; however, applications will need to transfer larger payloads occasionally -- for instance, for firmware updates. In contrast to HTTP, where TCP does the grunt work of segmenting and resequencing, CoAP is based on datagram transports such as UDP or Datagram Transport Layer Security (DTLS). These transports only offer fragmentation, which is even more problematic in constrained nodes and networks, limiting the maximum size of resource representations that can practically be transferred.

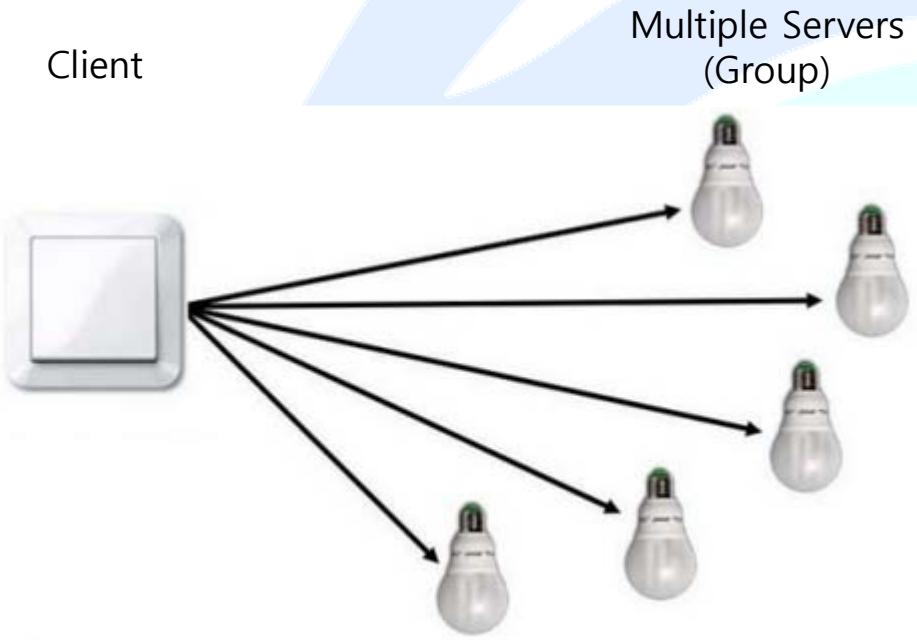
Instead of relying on IP fragmentation, this specification extends basic CoAP with a pair of "Block" options for transferring multiple blocks of information from a resource representation in multiple request-response pairs. In many important cases, the Block options enable a server to be truly stateless: the server can handle each block transfer separately, with no need for a connection setup or other server-side memory of previous block transfers. Essentially, the Block options provide a minimal way to transfer larger representations in a block-wise fashion.

A CoAP implementation that does not support these options generally is limited in the size of the representations that can be exchanged, so there is an expectation that the Block options will be widely used in CoAP implementations. Therefore, this specification updates RFC 7252.

HALLYM UNIVERSITY

Group Communication

- **Overview**
  - Multicast를 이용한 그룹 전송을 통해 복수 개의 기기를 동시에 제어
  - 응답이 필요한 경우 Unicast를 통해 개별적으로 Response를 수신
    - 충돌을 방지하기 위해 각 server들은 임의의 시간을 대기후 Response를 송신



Group Communication

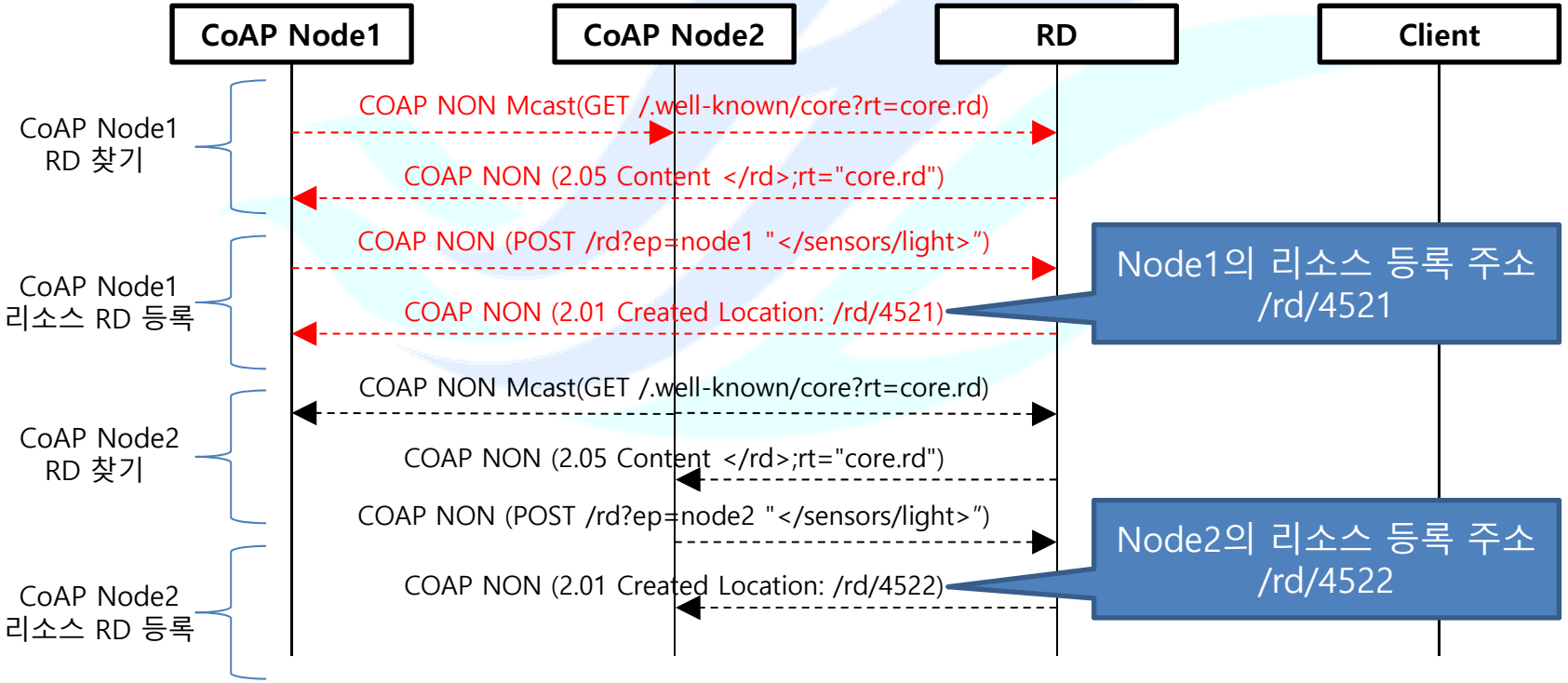
- **Group definition**
  - Group: Server들의 집합
  - Group의 멤버 설정
    - IP multicast 주소 또는 Hostname으로 미리 설정
    - Resource directory (RD)를 통해 설정
  - 그룹을 구성하는 Server들은 시간이 지남에 따라 동적으로 바뀔 수 있음
- **Group naming**
  - IP multicast address을 그대로 사용
    - Group의 IP 주소를 의미
  - Hostname을 사용
    - DNS (Domain Name System)가 지원될 때 사용

```
URI authority                                Targeted group of nodes
-------------------------------------------- -------------------------
all.bldg6.example.com                        "all nodes in building 6"
all.west.bldg6.example.com                   "all nodes in west wing,
                                              building 6"
all.floor1.west.bldg6.example.com            "all nodes in floor 1,
                                              west wing, building 6"
all.bu036.floor1.west.bldg6.example.com "all nodes in office bu036,
                                              floor 1, west wing,
                                              building 6"
```

HALLYM UNIVERSITY

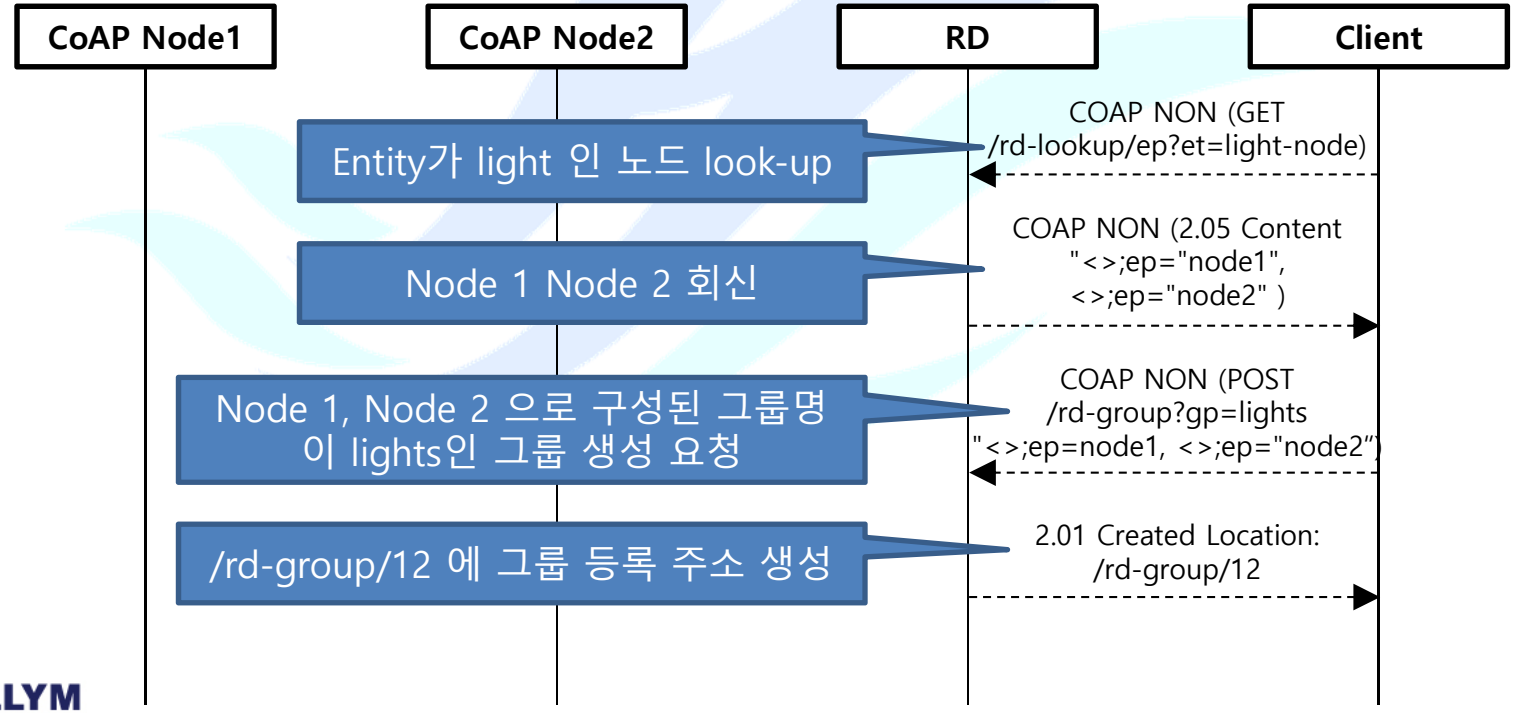
Group Communication

- **Example (1/3)**
  - Resource directory 찾기 및 리소스 등록
    - Server들은 전원 공급이 되면 Resource Directory (RD)를 찾고, 리소스를 등록함.
      - RD 찾기 요청 메소드: GET
      - 리소스 등록 요청 메소드: POST
      - RD 찾기 수행시 모든 CoAP 노드를 의미하는 multicast address (ff05::fd)를 사용



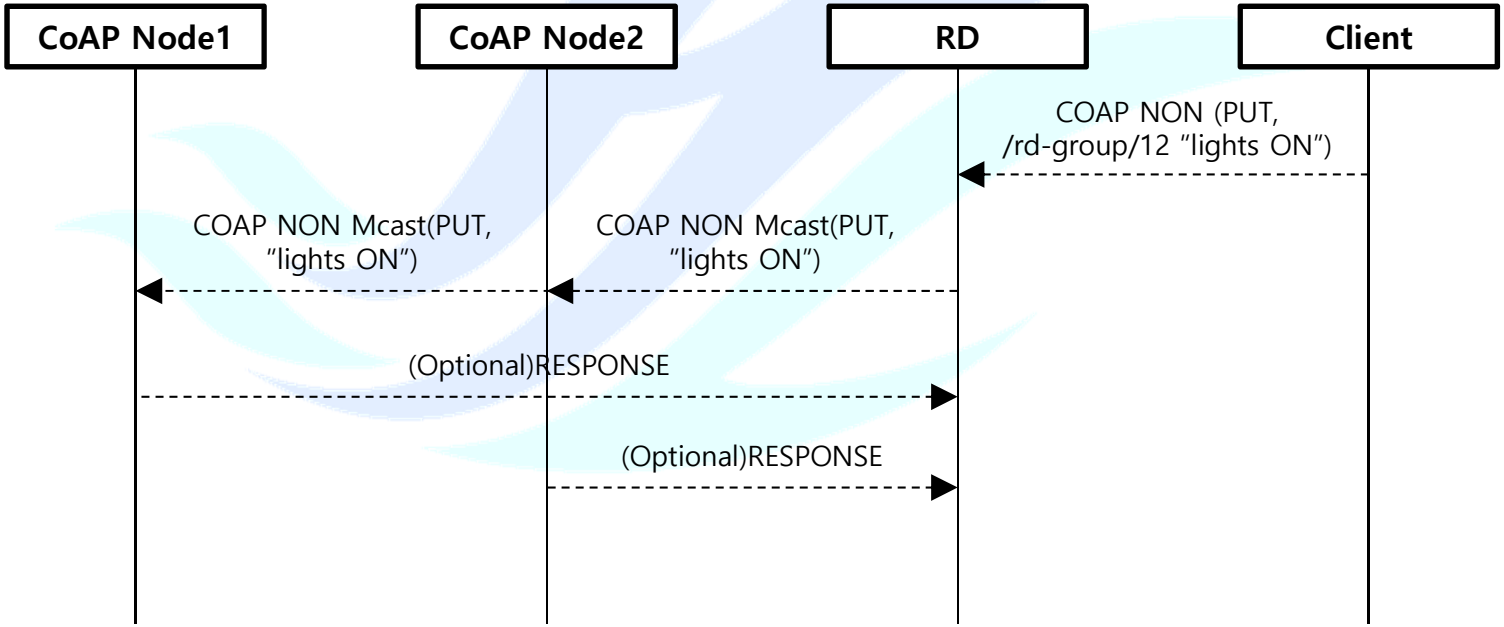
Group Communication

- **Example (2/3)**
  - Resource directory 내 리소스 찾기 (look-up) 및 그룹 등록
    - Client는 Look-up을 수행하여 RD 내 리소스를 찾고, Group 등록을 요청함.
      - 리소스 찾기 메소드: GET
      - 그룹 등록 요청 메소드: POST



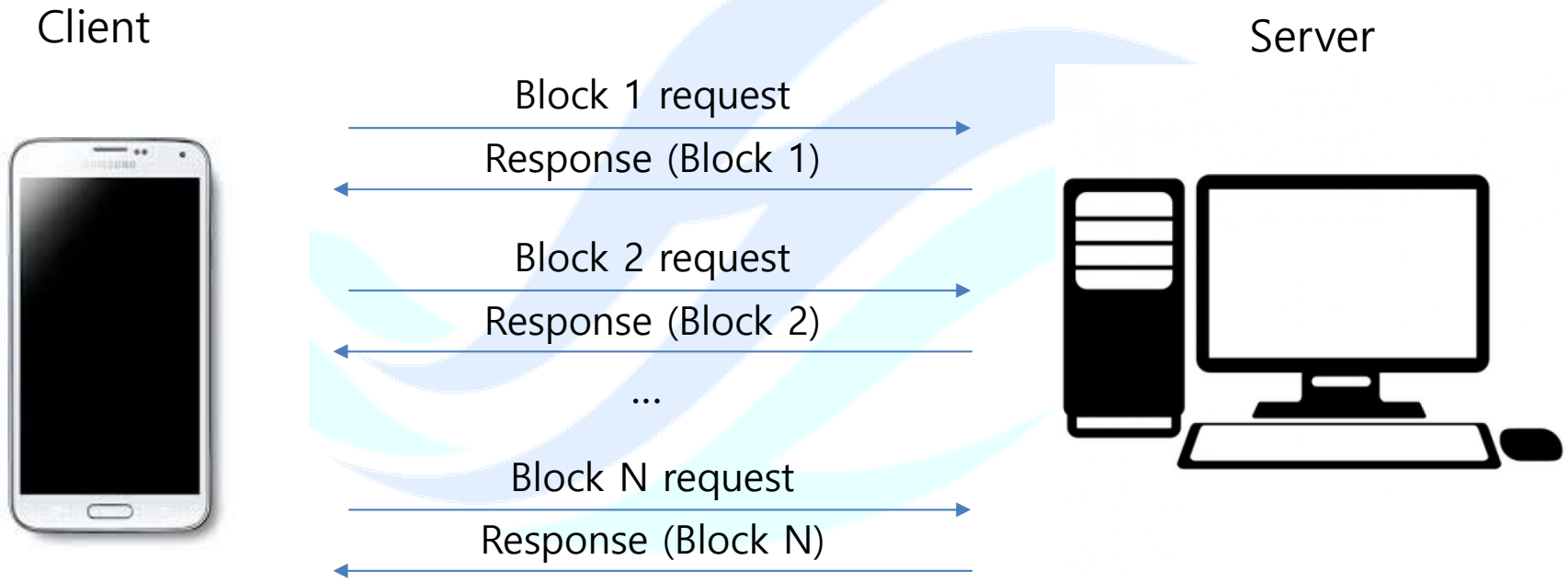
Group Communication

- **Example (3/3)**
  - 그룹 동작 요청
    - Client는
      - 그룹 동작 요청 메소드: PUT
      - 노드들의 응답은 필요시 수행할 수 있음.



Block-wise Transfer

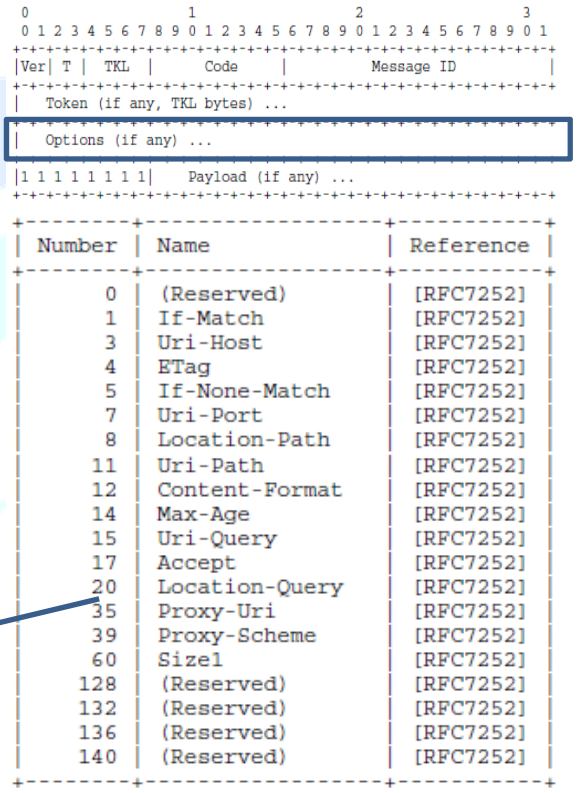
- **Overview**
  - 펌웨어 업데이트와 같은 큰 Payload를 전송할 때 사용



Block-wise Transfer

- **Block option**
  - Option No.: 23
    - Block Option 2: 응답 Message에 Payload 를 포함하는 경우에 사용 (앞 페이지 예시)
  - Option No.: 27
    - Block Option 1: 요청 Message에 Payload 를 포함하는 경우에 사용

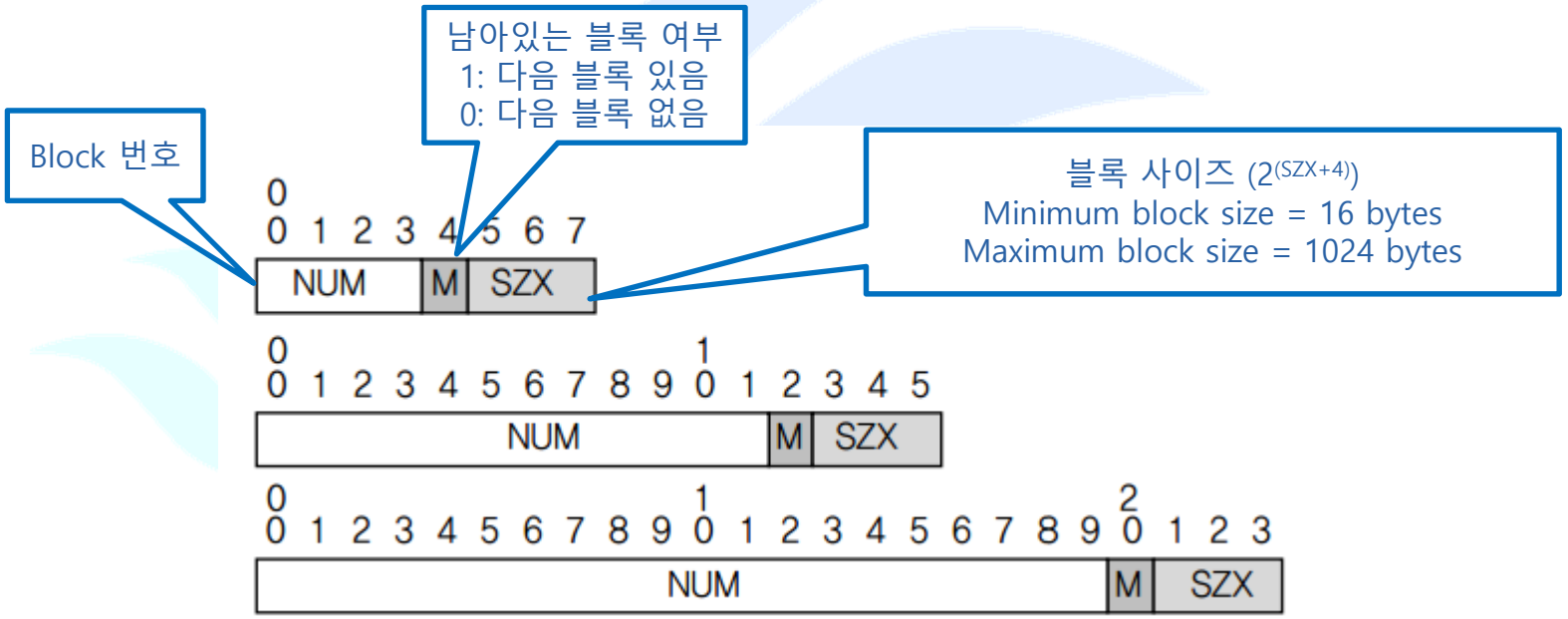
No.	Name	Format	Length	Default
23	Block2	uint	0-3 Byte	None
27	Block1	uint	0-3 Byte	None



Number	Name	Reference
0	(Reserved)	[RFC7252]
1	If-Match	[RFC7252]
3	Uri-Host	[RFC7252]
4	ETag	[RFC7252]
5	If-None-Match	[RFC7252]
7	Uri-Port	[RFC7252]
8	Location-Path	[RFC7252]
11	Uri-Path	[RFC7252]
12	Content-Format	[RFC7252]
14	Max-Age	[RFC7252]
15	Uri-Query	[RFC7252]
17	Accept	[RFC7252]
20	Location-Query	[RFC7252]
35	Proxy-Uri	[RFC7252]
39	Proxy-Scheme	[RFC7252]
60	Size1	[RFC7252]
128	(Reserved)	[RFC7252]
132	(Reserved)	[RFC7252]
136	(Reserved)	[RFC7252]
140	(Reserved)	[RFC7252]

Block-wise Transfer

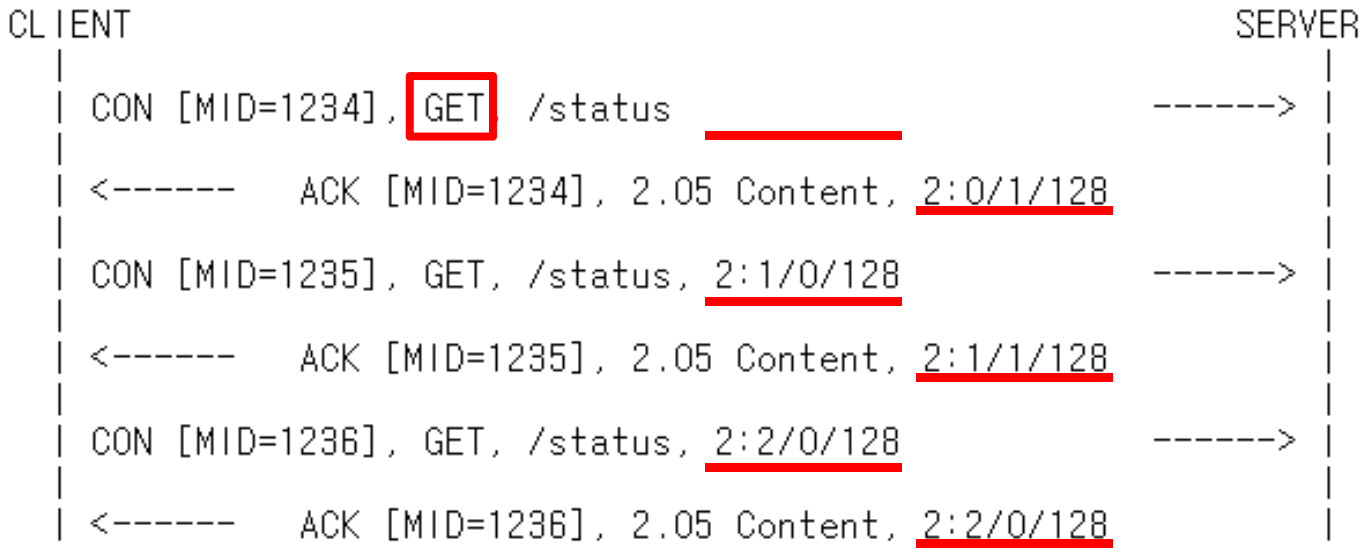
- **Block option field**
  - Block Option을 정의하여 블록 단위로 Payload를 전송할 수 있게 함
  - NUM, M, SZX로 구성되는 Block option을 정의함



블록 옵션 값 포맷

Block-wise Transfer

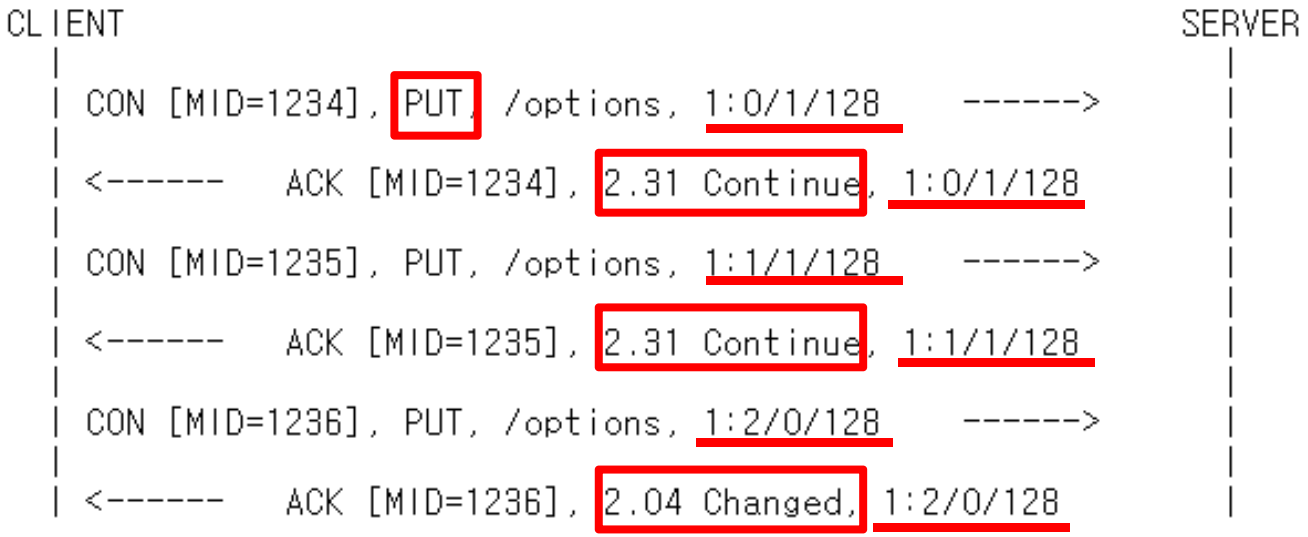
- **Example**
  - 콜론(:)의 앞의 숫자는 Block Option을 나타내며, 콜론(:) 다음의 숫자는 블록 번호, 슬래쉬(/) 다음은 M(more) 비트, 두 번째 슬래쉬(/) 다음의 숫자는 블록 사이즈를 나타냄
    - Block option : Block number / M / Block size
  - Server는 3개의 블록을 전송하고 있으며, 처음 두 블록의 Payload는 128바이트이고, 마지막 ACK의 payload는 1~128바이트 크기임



Block2 Examples

Block-wise Transfer

- **Example**
  - Client가 다수의 블록을 통해 Server에게 갱신을 요청함
    - 요청 메시지를 보낼 때 PUT 메소드를 사용하고, 블록 옵션 정보를 Payload와 함께 전송
    - 처음 2개의 요청 블록에 대한 ACK의 Response code 는 2.31 Continue로 뒤에 보낼 블록이 있음을 알림
    - 마지막 요청 블록에 대한 대한 ACK의 Response code 는 2.04 Changed로 성공적으로 갱신되었음을 알림



Block1 Examples