CoAP and LwM2M: Lightweight Protocols for Constrained IoT Devices

Victor Yang, binyu@wustl.edu
(A paper written under the guidance of Prof. Raj Jain)

Download 

Abstract

The Constrained Application Protocol (CoAP) and the Lightweight Machine-to-Machine (LwM2M) standard have taken a major position in the way many systems connect and handle constrained Internet of Things (IoT) end-devices. They are intended for use in bandwidth, memory, and battery constrained environments — secure communication, remote control, and fleet size and scale, are still anticipated. This paper examines the performance of CoAP and LwM2M in recent specifications and research directions. It considers the most important aspects of LwM2M 1.2, reviews work on LwM2M running over LTE-M and NB-IoT, evaluates the performance of CoAP and MQTT, analyses LoRaWAN and NB-IoT as base networks, and briefly assesses some specific deployments including IoT video, mobile robots, and adaptive management techniques. In each case the paper describes what has been done, what has been seen and what the results indicate for deployments in practice. We then combine these findings into a comparative discussion, which is frank about the benefits and drawbacks of CoAP and LwM2M. In the paper, it is concluded that faced with open challenges — including scalable firmware updates, key management, behavior over low-power wide-area networks and integration with more recent architectures — CoAP and LwM2M provide both good practical and technically feasible solutions for constrained IoT systems if the methods are put to good use.

Keywords: IoT, CoAP, LwM2M, LPWAN, MQTT, device management, performance evaluation, future directions.

Introduction

Most of the story of IoT depends on these very, very small and very, very cheap devices. Most come equipped with only a minuscule microcontroller, a handful of kilobytes or megabytes of memory and a low-powered radio. They sleep most of the time. But for these devices, we still want to do big-system tasks: authenticate them, configure things remotely, take measurements, receive alerts and occasionally update the firmware without ever having to go to front-of-house to examine the hardware. Even for classical web protocols like HTTP over TCP, the cost of latency, bytes, and energy needed can prove too high to justify in monetary terms. CoAP was developed out of this gap. What it does is keep its fundamental concept of RESTful communication (clients will have access to resources and talk to them in known ways) — but put it in something that the constrained device and a lossy network can handle. It takes over UDP instead of TCP, the headers are little binary keys, and relies on a fairly straightforward model of reliability one built with acknowledgement and retransmission (not with an entire stream abstraction). It also conforms

feature mapping with common IoT patterns as follows: the Observe method to make updates independent of the events and blockwise transfers for the larger payloads. LwM2M on top of CoAP, addresses in a single question: how can we systematically organize device management to be consistent enough to allow managing devices from multiple vendors on one platform. It defines a single object model for devices, configuration, connectivity and firmware and establishes collaboration between device (the LwM2M client) and management platform (the LwM2M server) through CoAP. More recent updates such as LwM2M 1.2 offer a more compact presentation of our data, multi-transport support, higher robustness and flexibility protection. In this paper, we consider CoAP and LwM2M from a few selected standards and literature around one good question-are these protocols a realistic option for constrained IoT devices today, and does it work when the context changes? To preserve the length yet with the essential details required for this course, this discussion is organized into a concise sequence of parts.

CoAP and LwM2M: Core Concepts

CoAP is better defined under a heading like “HTTP for constrained environments,” but that description only holds true as long as we pay attention to what it discards. Where texts are used for headers, CoAP has a four-byte base header and a small set of choices. Using it makes it clear that instead of relying on some tacit safe stream, it deals with each message in an absolute way, letting the sender determine whether or not a message is confirmable (and acknowledged) or non-confirmable. It’s a far better choice for the sleepy devices that wake up only for seconds and sends what they need to send, then fall back asleep without a link. The protocol further recognizes that the constant polling is wasteful. The Observe extension allows a client to request the server update when a resource change occurs so that updating should be pushed only when that service decides to upgrade. Likewise, blockwise transfers are used to securely allow devices to send large-sized messages, whether images from firmware or much more, in small pieces. But thanks to these two things alone CoAP is so naturally integrated with IoT that eliminating HTTP headers is not enough. LwM2M converts it to a CM-F based on CoAP message layer. Every device consists of "objects" - each one has instances and resources. Items are things like “Device,” “Firmware Update,” or “Connectivity Monitoring.” Queries and manipulations of these resources can occur through CoAP-like server operations. It defines basic flows for bootstrap (informing the device to whom you will talk using which keys), registration (telling the server what the device supports), continuous control and an information reporting method (frequently Observe). There is a general improvement in LwM2M 1.2, providing efficient encodings like SenML with CBOR, outlining the way to change notifications with attributes, extending transport options, and integrating with DTLS 1.3 and OSCORE. On a macro level, CoAP provides lean transport and interaction model, whereas LwM2M is a common vocabulary and lifecycle.

LwM2M over LTE-M and NB-IoT

One of the most natural tests of LwM2M being “lightweight enough” is to watch its performance using commercial cellular IoT networks like LTE-M and NB-IoT. This research uses one of the main studies in the scope and performed on the real devices of LwM2M client and executed it over these networks with simple but usual scenarios like recurrent data reporting, remote reads and writes, and simple notification patterns. The findings are basic though they are impactful. Most of

the latency observed in both experiments was not due to LwM2M processing, but rather a reflection of the underlying radio behavior, including paging cycles, sleep modes and coverage. For comparison, when comparing the protocols, LwM2M over CoAP always takes fewer bytes than MQTT over TCP when it comes to managing the equivalent management exchanges. NB-IoT and LTE-M are charged based on the number of data in and out, and energy sensitivity makes such a huge difference. The study also notes being able to implement the correct LwM2M right does not happen lightly, however, that after such a preliminary implementation you could achieve the final device for application on cellular IoT at no unnecessary trouble. Indeed, running LwM2M on top of CoAP doesn't "break" these networks; it works within their constraints and gets the standardization benefits it promises to deliver.

What LwM2M 1.2 Adds

Several such studies are based on the LwM2M 1.2 specification which provides specification of the features. The core specification follows the object and resource model and extends the family of supported encodings. SenML with CBOR keeps sensor payloads small without sacrificing the metadata you require. Not a sexy gimmick, but the saved bytes quietly prolong battery life and decrease the cost of data down over your life of a deployment. Even notification behavior has been defined better. If updates are sent, and for a given time interval, then that will depend more on how many times they are sent and what the conditions are. Instead of reporting any small variation, a device is set up to update if a value exceeds a certain threshold or the minimum time interval has passed. This better aligns with the random bursty nature of a lot of IoT signals. In other words, the transport and security specification is all about flexibility. It also confirms the classic CoAP-over-UDP with DTLS model and demonstrates how it is possible to use LwM2M over CoAP/TCP, plain HTTP, or MQTT. It also introduces modern security protocols, and introduces the OSCORE to safeguard end-to-end against proxies. We want system designers to be enabled to select combinations based on their deployment as long as the semantics of LwM2M is retained. Collectively these updates illustrate a protocol adapting to real-world deployments: more effective encodings; more nuanced control of reporting; differing approaches to endure different network and security conditions.

Table 1. LwM2M 1.2: Major Enhancements

Feature Area	Earlier Versions (1.0/1.1)	LwM2M 1.2 Improvement	Benefit
Data Encoding	TLV only	SenML + CBOR	Smaller, efficient payloads
Notification Model	Basic Observe	Min/Max time, thresholds, conditional triggers	Less noise, more relevant reporting
Network Transport	UDP/DTLS	CoAP/TCP, MQTT, HTTP	Higher flexibility across networks
Security	DTLS 1.2	DTLS 1.3, OSCORE	Stronger security, end-to-end protection
Resource Model	Static objects	More flexible metadata & object evolution	Better device representation

CoAP vs. MQTT

Real head to head tests matter because CoAP and MQTT can't be overlooked on IoT, they're everywhere with them. In terms of production output, CoAP's performance has generally been better in small request–response communication and single-device telemetry. Tiny headers plus UDP—without the TCP handshake—means fewer bytes of stuff on the wire, and (again more often than not) latency is marginally smaller! By contrast, MQTT is based on the idea of broker and publish–subscribe topics. Such an architecture carries considerable weight when both producers and consumers are involved in the process, or when message routing rules need to be followed, rather than efficiency per device being of a secondary concern. The downside of such a system is that the dependence on TCP and longer design process on MQTT can be cumbersome for devices that wake up slowly, or which transmit virtually no data at all. To add security will not seriously alter these factors of life. Both protocols do take up some time using DTLS or TLS, especially during handshakes, but CoAP generally acts as a low-skilled backbone with light transmitted demand for constrained-style workloads. This solidifies LwM2M's use of CoAP as the initial layer, and should further enable MQTT mappings based on its integration to broker-centric systems.

Table 2. CoAP vs MQTT: Key Differences

Category	CoAP	MQTT
Transport Layer	UDP	TCP
Message Pattern	REST (Request/Response, Observe)	Publish/Subscribe via Broker
Header Size	Very small (4-byte base)	Larger (TCP + MQTT header)
Connection Setup	None (no handshake)	TCP 3-way handshake + MQTT CONNECT
Best Use Case	Small, infrequent telemetry; sleepy devices	Multi-producer/multi-consumer messaging
Suited for Constrained Devices?	Excellent	Moderate
Latency	Usually lower	Usually higher due to TCP

Underlying Networks: LoRaWAN and NB-IoT

Whether CoAP and LwM2M are good ideas or not also relies on what gets crossed. Comparing LoRaWAN and NB-IoT, however, could be seen how tightly the link layer is guiding reality. As NB-IoT and LTE-M are based on licensed spectrum and scheduled access, they offer comparably consistent two-way communication while processing control messages, not just sensor uplinks. That profile meets the needs of LwM2M quite nicely: hardware can register, receive configuration writes, download firmware (if slow), etc. LoRaWAN offers amazing range with very low energy usage, but a limited and heavily regulated downlink capacity. It is very good for sparse and almost only uplink applications. Some forms of LwM2M or CoAP-style patterns do allow the gateway in

between, but sophisticated device management or large firmware updates are less than good. The takeaway is straightforward: CoAP and LwM2M are perfectly suited to technologies like NB-IoT and LTE-M. The devices may be engineered to work with LoRaWAN in limited ways, but tradeoffs and intricate engineering are often necessary.

Table 3. Suitability of CoAP/LwM2M Across IoT Networks

Feature	LTE-M	NB-IoT	LoRaWAN
Spectrum	Licensed	Licensed	Unlicensed
Downlink Capacity	High	Medium	Very limited
Latency	Low	Medium	High
Supports Device Management?	Yes	Yes	Partially; limited downlink
Suitable for Firmware Updates?	Yes	Yes	Poor
Best Fit for LwM2M	Excellent	Excellent	Limited (uplink-heavy only)

Extensions and Applicability Cases

Some of the cited works refer to specific cases of CoAP that have applications of CoAP in other industries which could indirectly validate CoAP design. VE-CoAP suggests modifications towards streamlining the streaming of IoT video such as parallel flows and unequal loss protection. This is only video-related information, but CoAP's message model can adapt to higher traffic density quite flexibly, but it doesn't pack the load as full HTTP does. Here is an example where we use CoAP in a mobile robotic system and compare it vs. HTTP, MQTT etc. The results show CoAP can be utilized for faster and suitable communication in a robot telemetry and control task when sensor dynamics and link control variables evolve. This means that CoAP is not restricted to very slow or simple devices, but can even perform a dynamic (and often not) application when it is selected well. Dynamic management scheme analysis identifies changes such as reporting intervals and encodings to real-time circumstances, where adjustments improve status. LwM2M already exposes many of these properties and is therefore ideally placed to address such adaptive schemes. All three of these papers are purposely short but, in aggregate, suggest that CoAP and LwM2M are not a brittle one-trick remedy. They are at the right level to be scaled up to an advanced level.

Assessment

Tying it all together makes CoAP and LwM2M appear to be sensible solutions to the reality of constrained IoT. Because of the lightweight, UDP-enabled design of CoAP, along with its support for Observe and blockwise transfers, its characteristics are more well-matched for communication requirements of large-scale IoT environments compared to the heavy machinery of HTTP. This is where LwM2M — a type of augmented version — comes in, a one-size-fits-all solution that offers users and operators a familiar path to implement device onboarding, configuration, telemetry and firmware updates without needing to reinvent those functions in different lines. The above studies empirically support these choices. LwM2M has an overhead for LTE-M and NB-IoT which is less than the required overhead of the underlying network. CoAP may be better suited for many small message solutions than MQTT but will not eliminate MQTT for our brokered architecture. If

adaptive schemes and specialized extensions are done, it looks like the protocol family can be shaped out for more demanding requirements. At the same time, however, the protocols are not optimal everywhere. Ecosystems oriented around MQTT and powerful devices can make LwM2M an overkill. The true functionality of CoAP/LwM2M cannot be realized, but rather inside very tight LPWAN environments. The key is that these technologies are best on the case when the goal is to organize and integrate many constrained devices into a common, interoperable environment.

Open Challenges

Still, we have several problems before CoAP and LwM2M might be considered a full solution for all constrained IoT instances. Yet another challenge is secure and scalable firmware update at fleet-scale. The firmware object and blockwise transfers in CoAP and LwM2M do the leg work when needed, but the task of coordinating updates without overwhelming the network or draining your batteries is a tough job. The use of key management is another area where theory is far from reality. Key provisioning and key rotation for millions of devices — some on barely ever or limited-band access links — is operationally messy. Though LwM2M and the underlying security protocols are secure channels, the surrounding environment still entails more sophisticated tooling and deployment protocols. It is still equally hard to operate over very rigid networks, especially for some classes of LPWAN. It is imperative for designers to have a balance between message size, reporting frequency, and reliability settings to avoid congestion or other regulatory problems. The standards are flexible, but not all mixes were extensively tested in real implementation scale. As ideas such as digital twins and semantic device descriptions take the spotlight, there is ongoing work to elegantly implement LwM2M-driven devices into those systems without forfeiting the lightweight benefits of the protocol.

Conclusion

CoAP, and also LwM2M have had their task clearly defined: communicating to very small devices — a process efficiently, securely, at scale. The work we summarize here indicates they tend to at least be able to accomplish that, or, at least, in a specific context. CoAP gets it all in one go, skimming the protocol fat and running as an online communication model. LwM2M uses the object-managing layer to fulfill real needs from boot-up to firmware updates. The literature, from real-world network measurements to comparisons with MQTT as well as the radio technology behind those applications and more real applications-domain cases, points in the same direction. Judiciously, given the limited network and the constraints of end devices, CoAP and LwM2M offer a strong, realistic foundation for constrained IoT systems. They are not magic, and do not eliminate the essence of good engineering judgment, but they are suitable for the problems they are intended to solve.

Acronyms

IoT: Internet of Things
CoAP: Constrained Application Protocol
LwM2M: Lightweight Machine-to-Machine (OMA LwM2M)
M2M: Machine-to-Machine

LPWAN: Low-Power Wide-Area Network
FOTA: Firmware Over The Air (firmware update)
DTLS: Datagram Transport Layer Security
TLS: Transport Layer Security
OSCORE: Object Security for Constrained RESTful Environments
MQTT: Message Queuing Telemetry Transport
SNMP: Simple Network Management Protocol
JSON/CBOR/SenML: Data encoding formats (JavaScript Object Notation / Concise Binary Object Representation / Sensor Measurement Lists)
NB-IoT: Narrowband IoT (cellular LPWAN)
LTE-M: LTE Cat-M (cellular LPWAN)
6LoWPAN: IPv6 over Low-Power Wireless Personal Area Networks
SCHC: Static Context Header Compression (for LPWAN)

References

OMA SpecWorks, "Lightweight M2M (LwM2M) v1.2: Core Specification," 2020 (rev. 2022).

OMA SpecWorks, "Lightweight M2M (LwM2M) v1.2: Transport Bindings," 2020.

Shelby, Z., Hartke, K., and Bormann, C., "The Constrained Application Protocol (CoAP)," RFC 7252, IETF, 2014.

Dvořák, R. et al., "LwM2M for Cellular IoT: Protocol Implementation and Performance Evaluation," ICUMT, 2023.

Seoane, V. et al., "Performance Evaluation of the CoAP Protocol with Security Support for IoT Environments," Computer Networks, 197, 2021.

Lorenz, M. et al., "Comprehensive Assessment of LoRaWAN and NB-IoT Performance Metrics," IEEE Internet of Things Journal, 2022.

Ghotbou, A. and Khansari, M., "VE-CoAP: A CoAP-based Protocol for IoT Video Transmission," Journal of Network and Computer Applications, 2021.

Sarkar, C. et al., "Development of CoAP Protocol for Communication in Mobile Robotic Systems Using IoT Technique," Scientific Reports, 2025.

Said, O., "A Dynamic Management Scheme for Internet of Things (IoT) Environments: Simulation and Performance Evaluation," Computers, Modeling and Engineering Sciences, 2022.

"What's New in LwM2M 1.2," Developer/Practitioner Article, 2020.

Last modified on November 17, 2025
This and other papers on recent advances in networking are available online at
http://www.cse.wustl.edu/~jain/cse5700-25/index.html
Back to Raj Jain's Home Page