Computer Science and Application 计算机科学与应用, 2026, 16(6), 16-23
Published Online June 2026 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2026.166205

Hans汉斯

面向物联网设备的高效可靠OTA更新协议设计

韩德川

中国电子科技集团公司第二十研究所，陕西 西安

收稿日期：2026年5月3日；录用日期：2026年6月3日；发布日期：2026年6月10日

摘　要

针对物联网设备软件在线升级过程中采用“停止－等待”协议传输文件效率低下问题，本文提出一种双层升级方案：传输层采用基于批量下发与接收方主动错误申请的选择性重传机制；存储层设计适配裸闪存的A/B分区原子更新架构。仿真采用马尔可夫突发丢包模型，对比传统停止－等待协议、CoAP分块传输与本文协议。结果表明：在1%~5%丢包率下，本文协议相比传统方案耗时降低84.4%~84.6%；相比CoAP同样降低约84.4%~84.6%。在5%丢包率下，传统协议成功率88.8%，CoAP为88.5%，本文协议达99.1%。在5%突发丢包环境下，传统协议成功率72.3%，CoAP为71.9%，本文协议达97.8%。本文还从理论上阐述了固件签名验签及控制消息重传机制，增强了安全性及鲁棒性。该方案为资源受限物联网设备提供了高效可靠的升级解决方案。

关键词

软件升级，物联网，可靠传输，原子更新，嵌入式系统

Design of an Efficient and Reliable OTA Update Protocol for IoT Devices

Dechuan Han

The 20th Institute of China Electronics Technology Group Corporation, Xi’an Shaanxi

Received: May 3, 2026; accepted: June 3, 2026; published: June 10, 2026

Abstract

To address the low efficiency of the “stop-and-wait” protocol in over-the-air (OTA) upgrades for IoT devices, this paper proposes a two-layer upgrade scheme. The transport layer adopts a selective retransmission mechanism based on batch distribution and receiver-initiated error requests; the storage layer designs an A/B partition atomic update architecture adapted to raw flash memory. The simulation uses a Markov burst loss model and compares the traditional stop-and-wait protocol,

文章引用: 韩德川. 面向物联网设备的高效可靠 OTA 更新协议设计[J]. 计算机科学与应用, 2026, 16(6): 16-23.
DOI: 10.12677/csa.2026.166205

CoAP block-wise transfer, and the proposed protocol. Results show that under 1%~5% packet loss rates, the proposed protocol reduces upgrade time by 84.4%~84.6% compared with the traditional scheme, and by approximately 84.4%~84.6% compared with CoAP. At a 5% packet loss rate, the success rates are 88.8% (traditional), 88.5% (CoAP), and 99.1% (proposed). Under 5% burst loss, the success rates are 72.3% (traditional), 71.9% (CoAP), and 97.8% (proposed). This paper also theoretically elaborates on firmware signature verification and control message retransmission mechanisms, enhancing security and robustness. The proposed scheme provides an efficient and reliable upgrade solution for resource-constrained IoT devices.

Keywords

Software Update, Internet of Things (IoT), Reliable Transmission, Atomic Update, Embedded Systems

1. 引言

物联网设备在软件在线升级过程中常面临设备资源的有限性、无线链路的不稳定性[1]以及升级可靠性等诸多严苛要求。当前主流升级方案依赖经典的“停止–等待”协议，即单包数据确认应答后继续发送下包数据，导致大量时间消耗在往返延迟上。链路不稳定出现丢包时触发超时重传机制，升级时间显著延长，甚至导致升级失败。物联网设备升级除传统“停止–等待”协议外，CoAP、LwM2M 以及 MQTT 等方法也被经常使用。CoAP 允许将大型固件拆分为多个块进行传输。每个块的接收方需发送确认消息，发送方超时后重传该块。CoAP 的“请求–确认”模式在半可靠信道上能够有效工作，但其本质仍是每块级别的等待确认，当单块尺寸较小时，往返延迟累积效应显著；当单块尺寸较大时，抗丢包能力下降。此外，CoAP 协议栈相对复杂，在 8 位 MCU 上完整实现存在一定挑战。LwM2M 主要针对单一设备的会话管理进行优化，在批量并发升级场景下，服务器需维护大量会话状态，资源开销较大。其重传策略继承自 CoAP，仍然无法避免每块等待确认的固有延迟。MQTT 作为发布/订阅协议，通过 QoS 级别提供不同程度的可靠性。QoS 2 (恰好一次)使用了四次握手，能够保证消息不重不丢，但在固件传输场景下，每个数据包都需要多次交互，开销巨大。同时，MQTT 面向松散耦合的消息而非有序文件流，实现固件完整重组需要额外定序与缓存逻辑，增加了终端复杂度，物联网 OTA 传输协议特性对比如表 1 所示。

尽管选择性重传在互联网领域已得到较好验证[2]，但其实现复杂且很难直接移植到资源受限的嵌入式平台。同时，现有研究多集中于对传输效率的优化，未能充分考虑 OTA 升级不仅包含对数据的传输要求，同时还必须考虑其与存储安全更新的协同设计，确保升级过程的原子性，防止设备升级失败从而“变砖”。

为此，本文提出一种轻量级的端到端升级方案，主要贡献包括：

1) 设计适用于嵌入式场景的轻量级选择性重传协议，通过批量下发与主动错误申请机制提升传输效率；

2) 提出与传输协议协同的存储更新架构，确保从网络到存储的全链路可靠性；

3) 采用突发丢包模型，搭建离散事件仿真实验，与传统方法及 CoAP 方法进行对比，验证方案有效性；

4) 从理论上补充固件签名验签与控制消息可靠传输机制，提升方案的完整性和鲁棒性。

Table 1. Comparison of IoT OTA transmission protocol characteristics
表 1. 物联网 OTA 传输协议特性对比

特性	传统停止－等待	CoAP 分块传输	MQTT QoS 2	本文协议
传输范式	单包确认	块级请求/确认	发布/订阅 ＋4 次握手	批量下发 ＋ 主动纠错
重传机制	超时重传(每包)	超时重传(每块)	接收端未确认即重传	选择性重传(接收端主动申请)
协议状态复杂度	极低	低	中	低(仅需维护位图)
往返延迟影响	极大	较大	中	极小(仅纠错阶段等待)
与存储原子性的协同	无	无	无	有(外部缓存 ＋A/B 分区)
典型适用场景	理想信道、极简系统	中低丢包率、小文件	消息可靠性高	资源受限、高丢包、中大型固件

2. 高效传输协议与存储架构协同设计

本方案由高效传输协议与安全存储架构两部分组成，两者协同工作确保升级的完整性与原子性。

2.1. 高效传输协议设计

本协议的设计以提升效率为核心，通过四个阶段的流水线操作，将传统的“发送－等待－确认”模式转变为高效的“批量下发－主动纠错”模式，其整体交互流程如图 1 所示。

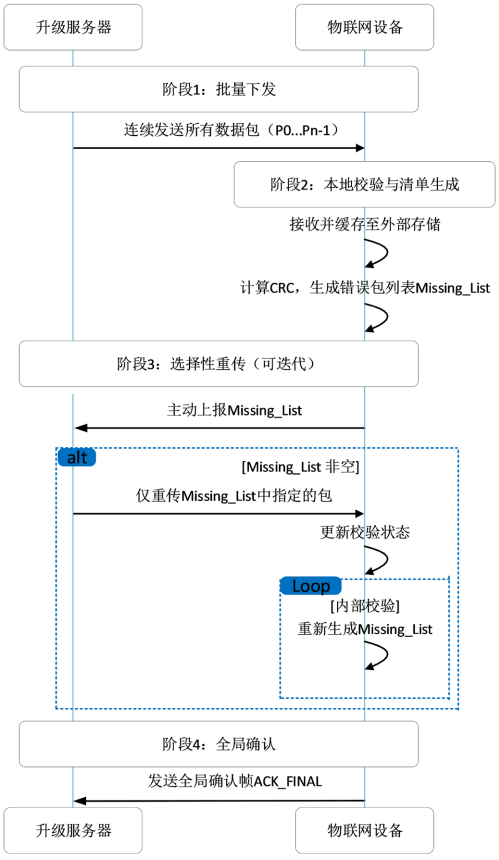

Figure 1. Interaction flowchart of the optimized transmission protocol
图 1. 优化传输协议交互流程图

优化传输协议包含以下四个主要交互过程：第一阶段，服务器将升级数据拆分成若干个数据包并将其连续发送给设备，期间无需等待设备应答确认报文，以充分利用网络带宽。第二阶段，设备在接收数据的同时开展 CRC 校验，记录错误数据包序号，待所有数据接收完毕后，生成错误数据包列表。第三阶段，设备主动向服务器发送错误数据包列表；服务器收到后仅针对列表中的错误包进行重传；第二阶段和第三阶段可重复迭代，直至所有数据都被正确接收。第四阶段，设备将所有数据校验通过后，发送特殊格式的全局确认帧，标志数据传输成功。该设计借鉴了 TCP 协议中 SACK 机制的思想[3]，但针对嵌入式设备资源受限的特点进行了轻量化适配。除数据类消息外，控制类消息采用确认与重传机制，在不显著增加复杂度的前提下确保控制消息的可靠送达，控制类消息保障机制与主动传输流程正交，不会影响批量下发阶段的效率，仅在异常发生时增加少量超时等待，总体开销可忽略。

该设计通过将多次零散的确认与重传，整合为少数几次集中式的校对与补发，大幅减少了通信交互次数和总耗时，有效提高了带宽的利用率。

2.2. 安全存储架构设计

为确保升级过程的原子性与安全性，本方案设计了与优化传输协议配合使用的存储架构，数据流图如图 2 所示。

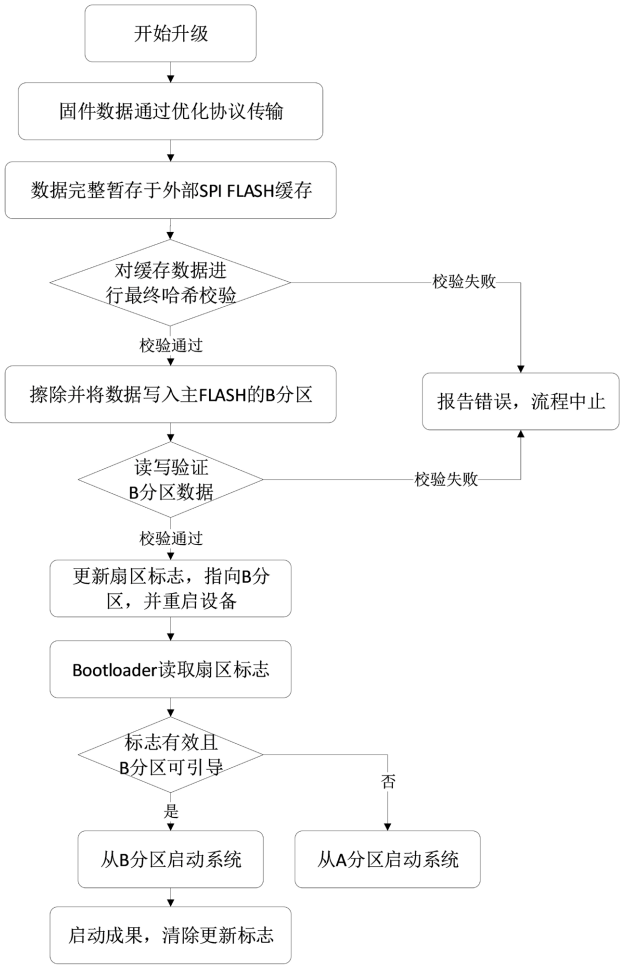

Figure 2. Data flow diagram of the atomic update storage architecture
图 2. 原子更新存储架构数据流图

该架构由 A/B 两个分区组成，将主闪存划分为两个完整的存储分区(A 和 B)，并引入外部串行闪存用于缓存接收数据。当升级数据包通过优化协议完整下载至外部缓存并校验无误后，系统将数据写入非活动分区(如 B 分区)，随后更新扇区启动标志以切换启动分区。设备重启后，引导程序根据扇区标志决定从 A 或 B 分区启动程序。若 B 分区程序引导失败，则回滚至 A 分区程序继续启动。此设计保证了即使在升级过程中发生断电或意外事故，设备始终可正常启动，从而避免了设备“变砖”的风险。

为防止固件被篡改或来自非法源头，所有升级包在服务端生成时，均附加其哈希值的数字签名。设备在完成所有数据包的下载与 CRC 校验后，从外部串行闪存中重组完整固件镜像，计算其哈希值，使用设备预置的公钥验证下载得到的固件签名。仅当验签通过后，方执行将数据从外部缓存拷贝至非活动分区的写入操作，若验签失败，立即丢弃缓存数据，向服务器报告“固件无效”错误，并终止升级流程，维持原分区启动。

此架构与优化后的传输协议自然契合，外部缓存同时服务于协议纠错与数据暂存，实现了传输与存储的安全解耦。

3. 理论分析与性能建模

为量化评估协议性能，本节建立理论模型。定义以下参数：

N：总数据包数量(固件 512KB，每包 512 字节，故 $N=1024$)。

T_{tx}：单包传输时间。

T_{rtt}：网络往返延迟(传统协议中为等待 ACK 的时间，新协议中为发起错误申请到开始重传的延迟)。

P：随机独立丢包率。

3.1. 传统协议耗时模型

传统协议为停止–等待模式，其理想情况(0%丢包)下的耗时为基础模型：

$$T_{legacy_{base}} = N \times \left(T_{tx} + T_{rtt}\right)$$

存在丢包时，每次丢包将引入至少一次额外的超时等待与重传。近似认为，总耗时随丢包率 p 呈指数增长趋势，可表达为：

$$T_{legacy} \approx N \times \left(T_{tx} + T_{rtt}\right) / \left(1-p\right) + \omega\left(p\right)$$

其中 $\omega\left(p\right)$ 为高次丢包引发的复杂重传叠加时间。CoAP 由于头部开销略大，实际耗时会比传统协议稍高(约 1%~2%)，为简化模型，后续分析中视两者基本相等。

3.2. 新协议耗时模型

新协议耗时分为两个阶段：

1) 无确认批量下发阶段：耗时固定为 $N \times T_{tx}$。

2) 选择性重传阶段：该阶段耗时取决于丢包分布。假设平均每轮重传需纠正 M 个包，进行 K 轮，则总耗时约为：

$$T_{proposed} \approx N \times T_{tx} + K \times \left(T_{rtt} + M \times T_{tx}\right)$$

在理想信道下，$K=0$，因此：

$$T_{proposed} = N \times T_{tx}$$

效率显著优于传统协议。

理论效率对比：

在0%丢包率下，定义效率提升比η为：

$$\begin{aligned}\eta &= 1 - T_{proposed} / T_{legacy_{base}} \\ &= 1 - \left(N \times T_{tx}\right) / \left(N \times \left(T_{tx} + T_{rtt}\right)\right) \\ &= T_{rtt} / \left(T_{tx} + T_{rtt}\right)\end{aligned}$$

当T_{rtt}远大于T_{tx}时，η趋近于1，即新协议可消除绝大多数由等待应答引入的无效时间。

4. 仿真验证与性能分析

为验证高效传输协议与安全存储架构的性能，本文在Visual Studio 2017环境下使用C语言开发了离散事件仿真模型，精确模拟传统协议、CoAP分块传输协议与优化协议在几种典型丢包场景下的软件远程升级情况。

4.1. 仿真环境与方法

仿真模型采用事件驱动架构，包含以下核心模块：

网络信道模型：实现了两种模型：(a) 随机独立丢包模型；(b) 两状态马尔可夫链(Gilbert-Elliott)模型，用于模拟突发丢包。马尔可夫链参数设置为：好状态丢包率 0.1%，坏状态丢包率20%，平均好状态驻留长度1000包，平均坏状态驻留长度10包。同时引入了延迟抖动，T_{rtt}服从均值为50 ms、标准差为10 ms的正态分布。

协议栈模块：分别实现了传统停止－等待协议(每包应答，超时重传 3 次)、基于 CoAP Block-Wise Transfer的参考实现(块大小512字节，超时重传3次)以及本文提出的新协议(批量下发－主动错误申请，并增加了控制消息重传机制)。

统计模块：记录每次仿真的总耗时、成功率及资源使用情况。

仿真参数设置如下：

固件大小：512 KB，分包为1024个数据包(每包512字节)。

时间参数：单包传输时间$T_{tx} = 10$ ms，网络往返延迟$T_{rtt} = 50$ ms。

网络环境：分别模拟0%、1%、3%、5%四种随机独立丢包率。

仿真方法：每种场景独立运行1000次，取统计平均值。

资源模型：为设备节点设置有限的内存缓存(32KB)，模拟资源受限环境。

4.2. 仿真结果与分析

4.2.1. 随机独立丢包模型仿真结果

1) 升级耗时对比

仿真耗时结果如表2所示。从表中数据可以看出，传统协议与CoAP耗时非常接近(CoAP因头部开销略高，耗时约多1%)。新协议相比两者耗时降低84.4%~84.6%，且随丢包率上升耗时增长极小。

Table 2. Comparison of upgrade time in simulation (random independent packet loss, unit: s)
表2. 升级耗时仿真对比(随机独立丢包，单位：秒)

丢包率	传统停止－等待协议	CoAP分块(512B)	新协议	新协议 vs 传统	新协议 vs CoAP
0%	66.56	67.12	10.24	−84.6%	−84.7%
1%	67.19	67.78	10.40	−84.5%	−84.7%

续表

3%	**68.45**	**69.09**	**10.64**	**−84.5%**	**−84.6%**
5%	69.79	70.46	10.88	−84.4%	−84.6%

2) 升级成功率对比

升级成功率对比结果如表 3 所示。传统协议成功率在 5%丢包率下为 88.8%；CoAP 因头部开销导致有效载荷占比略低，重传成功率稍差(88.5%)；新协议成功率高达 99.1%。

Table 3. Comparison of upgrade success rates
表 3. 升级成功率对比

丢包率	传统停止－等待	CoAP 分块(512B)	新协议
0%	100.00%	100.00%	100.00%
1%	99.80%	99.70%	100.00%
3%	97.10%	96.90%	99.90%
5%	88.80%	88.50%	99.10%

4.2.2. 突发丢包与延迟抖动模型仿真结果

采用 Gilbert-Elliott 突发丢包模型，平均丢包率 5%，结果如表 4 所示。突发丢包使传统协议和 CoAP 性能急剧恶化(成功率降至 72.3%和 71.9%)，而新协议仍保持 97.8%的高成功率，耗时仅小幅增加。

Table 4. Performance comparison under burst packet loss and delay jitter (average packet loss rate 5%)
表 4. 突发丢包与延迟抖动环境下的性能对比(平均丢包率 5%)

协议方案	平均耗时 (s)	耗时标准差 (s)	成功率
传统停止－等待	125.6	32.5	72.3%
CoAP 分块(512B)	128.3	33.1	71.9%
本文协议	14.2	2.3	97.8%

4.2.3. 资源开销分析

新协议的实现带来了可控的额外资源开销。根据仿真结果，用于维护协议状态(接受位图和错误列表)的内存开销在峰值时增加约为 6KB，同时，为了实现 CRC 的计算以及列表操作，CPU 的使用开销在峰值时预计会增加约 3.5%。控制消息重传机制增加约 200 字节状态变量，签名验签需要公钥(64 字节)和哈希缓冲区(32 字节)。本方案的实现需增加外接串行闪存用于升级数据缓存，这种实现方式虽增加了物料成本，但有效避免了对设备稀缺片内 RAM 的占用，为资源有限的嵌入式设备的远程升级提供了可行性。

5. 讨论与结论

5.1. 方案优势总结

理论分析与仿真结果共同表明，本方案较传统停止－等待和 CoAP 分块传输在传输效率与可靠性方面均表现出较大优势。其设计核心源于架构层面的协同设计：传输层协议通过范式转变大幅避免了无效等待时间；存储层架构通过原子更新设计确保了数据写入的准确性，两者结合实现了端到端的优化。

效率方面：新协议在 0%~5%丢包率下耗时降低均保持在 84.4%以上，这主要得益于批量下发机制消

除了往返延迟的累积效应，选择性重传避免了冗余数据的大批量传输。

可靠性方面：新协议在 5%高丢包率下达到 99.10%的成功率，在突发丢包下仍能达到 97.8%，相比传统协议的 72.3%及 CoAP 分块传输的 71.9%均有显著提升。这源于全局重传机制将重传机会从单个数据包扩展至整个数据集，显著提高了容错能力。

5.2. 局限性及未来工作

本方案需在结构设计时增加串行闪存用于缓存升级数据，这无疑提高了设备的制造成本。在有限缓存条件下，将来可进一步探索研究自适应分块传输策略[4] [5]，将固件分块进行“传输–写入”流水线操作，在研制成本与传输效率之间寻找平衡。此外，当前仿真模型采用了随机独立丢包，未来可纳入突发丢包、时变延迟[6]等更复杂的网络模型以进一步验证协议的鲁棒性。下一步工作重点在于真实硬件平台上的原型实现与测试，以验证其在复杂真实环境下的性能与资源开销。

5.3. 结论

本文针对物联网设备的远程升级，提出并验证了一套完整的解决方案。通过轻量级选择性重传协议与原子存储更新架构的协同设计，有效解决了传统方案在效率与可靠性上的明显不足。仿真结果显示，该方案能大幅缩短软件升级时间，并在高丢包环境下及突发丢包环境下表现出极高的鲁棒性。本方案为物联网设备的大规模部署、可靠升级提供了设计参考。

参考文献

[1] 吴随愿. 面向联邦学习的智能工业物联网多维资源分配与优化[D]: [硕士学位论文]. 邯郸: 河北工程大学, 2023.

[2] 王晨光. 视频通信中的选择性重传技术[J]. 电子制作, 2014(2): 148.

[3] 吴昊. 网络包自动分析系统研究[D]: [硕士学位论文]. 华侨大学, 2022.

[4] 朱原玮. 面向沉浸式交互的扩展现实视频传输关键技术研究[D]: [博士学位论文]. 北京: 北京邮电大学, 2025.

[5] Wang, L., Li, C., Dai, W., *et al*. (2023) QoE-Driven Adaptive Streaming for Point Clouds. *IEEE Transactions on Multimedia*, **25**, 2543-2558. https://doi.org/10.1109/tmm.2022.3148585

[6] 燕晶. 基于端到端测量的网络拓扑发现研究[D]: [硕士学位论文]. 曲阜: 曲阜师范大学, 2024.