An Efficient Approach to Wireless Firmware Update Based on Erasure Correction Coding

50

Berk Kivilcim, Daniel Zhou, Zhijie Shi, and Kaleel Mahmood

Abstract

Updating the firmware of the deployed nodes in a large sensor network can be time consuming and challenging. One way to perform these updates is through an over-the-air (OTA) protocol that utilizes repeated rounds of firmware broadcasting. In this work, we propose to use erasure correction coding to improve the efficiency of the existing protocol. With the new method, receivers recover lost packets from received ones, instead of waiting for repeated transmissions. We implemented the proposed protocol and compared it with the existing method. Both theoretical analysis and experimental results demonstrate the advantages of the new approach. In our experiment, when the packet lost ratio was 27%, the new method achieved 99% success rate on firmware transmissions while the existing approach failed.

Keywords

Erasure coding · Firmware update · IEEE 802.15.4 · Internet of things · Reed-Solomon code · Sensor network · Wireless protocol

B. Kivilcim
Department of Computer Science, University of Southern Mississippi, Hattiesburg, MS, USA
e-mail: berk.kivilcim@usm.edu

D. Zhou
Letters and Sciences, University of Maryland, College Park, MD, USA
e-mail: dxzhou11@terpmail.umd.edu

Z. Shi (✉) · K. Mahmood
Department of Computer Science and Engineering, University of Connecticut, Storrs, CT, USA
e-mail: zshi@uconn.edu; kaleel.mahmood@uconn.edu

50.1 Introduction

Large scale wireless sensor networks are becoming increasingly more wide spread due to the growth of the Internet of things (IoT) and cyber-physical systems. They cover large physical areas and collect huge amounts of data, which are essential in many industrial monitoring and control systems.

One practical issue in large-scale sensor networks is firmware updates. Common practice is updating individual nodes manually via physical access. Recently, an Over the air (OTA) method was designed [1], but it still updates one node each time.

OTA broadcasting protocols that can update many nodes at the same time are inherently more efficient and increase the scalability of the system significantly. Such a protocol was developed in [2]. It is a one-way broadcasting protocol and allows multiple nodes to receive an update simultaneously from a transmitter node.

The wireless protocol in [2] is based on repeated transmissions. The TX repeats the transmission potentially many times. This method can be slow and waste bandwidth. In the worst case scenario, a receiver may keep missing the same packets in multiple rounds. Thus the receiver cannot perform a firmware update, despite repeatedly having received many correct packets.

In this paper we aim to improving the efficiency of the wireless firmware update protocol. We adopt the erasure correction coding approach where the receiver can recover the packets that are lost during transmission. In particular, we have implemented an erasure correction code based on a dual Reed-Solomon code over Galois field GF(2^8) [3]. Instead of sending out the same packets repeatedly, we encode a set of K packets into a new, larger set of N packets that are then transmitted. As long as a receiver receives at least K encoded packets, out of N packets, it can reconstruct the original K packets and then perform the firmware update. We conducted

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

S. Latifi (ed.), *ITNG 2023 20th International Conference on Information Technology-New Generations*, Advances in Intelligent Systems and Computing 1445,
https://doi.org/10.1007/978-3-031-28332-1_50

side-by-side comparison of the repetition and the coding approaches in experiments where multiple receivers are located at different distances from the transmitter. Both theoretical and experimental results show the superior performance of the coding approach.

Our contribution is to apply erasure correction coding to improve the efficiency of wireless broadcasting protocols for firmware updates.

The remainder of the chapter is structured as follows. Section 50.2 describes the proposed approach. Section 50.3 presents the experiment results. Section 50.4 summarizes the chapter and discusses possible improvements and future work.

50.2 Firmware Update Solutions

50.2.1 The Existing Approach

Figure 50.1 shows the existing approach in [2]. A firmware of 100 kilobytes is divided into $P = 25$ pages, with each page being 4 kB. Every page is divided into $K = 37$ blocks so that each block of 112 bytes can be carried in one packet. Each packet has a header that specifies the version number of the firmware, the page number in the firmware, and the packet number in a page. The receiver records the received packets and stores them based on the page number and the packet number. To ensure all K packets of an individual page are received correctly, the packets in a page are transmitted repeatedly M times. A receiver needs to receive all K packets in a page successfully to construct a page and receive all P pages in the firmware correctly to update the firmware.

Let p denote the Packet Loss Rate (PLR) at the communication link level. The probability of a successful firmware update (denoted by C) via the repetition approach (denoted by rep) is computed as [2].

$$\Pr(\mathrm{C};\mathrm{rep}) = (1-p^M)^{PK}. \tag{50.1}$$

With a given PLR, the success rate can be improved via increasing M, the transmission rounds. However, this method leads to higher costs, such as, more data transmission in the channel and longer transmission time.

50.2.2 The Proposed Approach

We propose to wirelessly transmit the firmware via an erasure coding approach. As in [2], we divide the firmware into P pages, and transmit the firmware page by page. Figure 50.2 shows the coded approach on transmission of each page. For each page, the K original packets are encoded into N coded packets, where $N > K$.

A linear (N, K) code can be specified via a generator matrix as:

$$\mathbf{c}_{N\times 1} = \mathbf{G}_{N\times K}\mathbf{m}_{K\times 1} \tag{50.2}$$

where $\mathbf{m}$ is the original block, $\mathbf{c}$ is the coded block and $\mathbf{G}$ is the generator matrix. The matrix-vector multiplication is performed on a finite field. In this paper, we design $\mathbf{G}$ based on a dual Reed-Solomon code over GF(2^8). Define α as a primitive element of GF(2^8), and denote $\alpha_i = \alpha^i$. The matrix $\mathbf{G}$ is constructed as:

$$\mathbf{G} = \begin{bmatrix} 1 & \alpha_0 & \alpha_0^2 & \cdots & \alpha_0^{K-1} \\ 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{K-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_{N-1} & \alpha_{N-1}^2 & \cdots & \alpha_{N-1}^{K-1} \end{bmatrix}_{N\times K} \tag{50.3}$$

The encoding and decoding of the K packets are done as follows. Let us label the $K = 37$ packets as:

- Packet 1: $x_1^{(1)}, .., x_1^{(L)}$
- Packet 2: $x_2^{(1)}, .., x_2^{(L)}$
- …
- Packet K: $x_K^{(1)}, .., x_K^{(L)}$

where L is the number of bytes in each packet. The encoding is done as follows.

$$\begin{bmatrix} y_1^{(1)} & \cdots & y_1^{(L)} \\ y_2^{(1)} & \vdots & y_2^{(L)} \\ \vdots & \vdots & \vdots \\ y_N^{(1)} & \cdots & y_N^{(L)} \end{bmatrix} = \mathbf{G} \begin{bmatrix} x_1^{(1)} & \cdots & x_1^{(L)} \\ x_2^{(1)} & \cdots & x_2^{(L)} \\ \vdots & \vdots & \vdots \\ x_K^{(1)} & \cdots & x_K^{(L)} \end{bmatrix} \tag{50.4}$$

The N packets that are transmitted wirelessly are:

- Packet 1: $y_1^{(1)}, .., y_1^{(L)}$
- Packet 2: $y_2^{(1)}, .., y_2^{(L)}$
- …
- Packet N: $y_N^{(1)}, .., y_N^{(L)}$

On the receiver side, a node may not receive all N packets correctly. However, as soon as a receiver receives K packets, with packet numbers identified as $\{i_1, \ldots, i_K\}$, it can reconstruct the original K packets. Let $\{y_i^{(1)}, ..., y_i^{(L)}\}$ denote the ith received packet, where $i \in \{i_1, ..., i_K\}$. Equation (50.4) suggests that

$$\begin{bmatrix} y_{i_1}^{(1)} & \cdots & y_{i_1}^{(L)} \\ y_{i_2}^{(1)} & \vdots & y_{i_2}^{(L)} \\ \vdots & \vdots & \vdots \\ y_{i_K}^{(1)} & \cdots & y_{i_K}^{(L)} \end{bmatrix} = \overline{\mathbf{G}} \begin{bmatrix} x_1^{(1)} & \cdots & x_1^{(L)} \\ x_2^{(1)} & \cdots & x_2^{(L)} \\ \vdots & \vdots & \vdots \\ x_K^{(1)} & \cdots & x_K^{(L)} \end{bmatrix}, \tag{50.5}$$

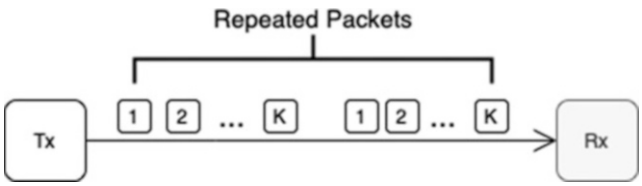

Fig. 50.1 Repeated approach, $K = 37$, $M = 2$

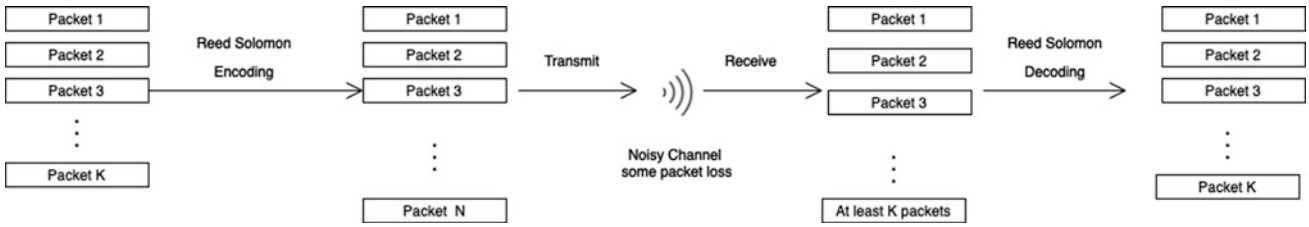

Fig. 50.2 Reed Solomon approach, $K = 37$, $N = 2K = 74$

where the matrix $\overline{\mathbf{G}}$ consists of K rows of $\mathbf{G}$, with row indexes of $\{i_1, \ldots, i_K\}$. Due to the Vandermonde structure of $\mathbf{G}$, any K rows of $\mathbf{G}$ are linearly independent and hence $\overline{\mathbf{G}}$ is guaranteed to have full rank. We can find the inverse of matrix $\overline{\mathbf{G}}$ via Gaussian elimination over GF(2^8). The original data packets are recovered by multiplying $(\overline{\mathbf{G}})^{-1}$ with the received packets as:

$$\begin{bmatrix} x_1^{(1)} & \cdots & x_1^{(L)} \\ x_2^{(1)} & \cdots & x_2^{(L)} \\ \vdots & \vdots & \vdots \\ x_K^{(1)} & \cdots & x_K^{(L)} \end{bmatrix} = (\overline{\mathbf{G}})^{-1} \begin{bmatrix} y_{i_1}^{(1)} & \cdots & y_{i_1}^{(L)} \\ y_{i_2}^{(1)} & \vdots & y_{i_2}^{(L)} \\ \vdots & \vdots & \vdots \\ y_{i_K}^{(1)} & \cdots & y_{i_K}^{(L)} \end{bmatrix} \quad (50.6)$$

A receiver receives a page successfully if it receives at least K, out of N, coded packets in the page correctly. And a firmware is correctly received only if all pages are received correctly. Hence, the probability of a successful firmware transmission via the coding protocol is:

$$\Pr(\text{C; coded}) = \left[\sum_{k=K}^{N} \binom{N}{k} (1-p)^k p^{N-k} \right]^P, \quad (50.7)$$

where p is the packet loss rate, K is the number of packets in a page, and N is the number of encoded packets.

50.2.3 Theoretical Comparison

To create a theoretical model, we compare the two approaches using Eqs. (50.1) and (50.6). We fix all of the values except for p, the packet loss rate. For the encoded protocol, we set N equal to $2K$. In the repetition protocol we set $M = 2$. So both approaches transmit the same number of packets.

By changing the value of p, we generate the curve shown in Fig. 50.3. It can be observed that the encoded protocol significantly outperforms the repetition protocol. The repetition protocol requires an extremely low packet loss rate to successfully transmit a firmware. The encoded protocol is much more robust and can reliably transmit the firmware at a packet loss rate up to about 40%.

Figure 50.3 highlights the benefit of the increased success rate when one transmitter is updating the firmware on one receiver. In large networks where many nodes need update simultaneously, the coding approach can significantly reduce the feedback information from the receiver nodes requesting the missing packets. In addition, a missing coded packet can be replaced by another coded packet as long as K different packets are received. In a multi-hop network, a node can also relay its received packets to its neighbors.

50.3 Implementation and Testing

Both the repetition and the coding protocols were implemented using TI LAUNCHXL-CC2652 LaunchPads. They ran in the debug mode using Code Composer Studio 11.2. To transmit and receive the packets, we used TI's EasyLink API [4]. We modified the example code for broadcasting to include both repetition and coding protocols. Figure 50.4 shows the experimental settings, where a transmitter node broadcasts firmware to three receiver nodes placed at distances of 5 m away, 11 m away, and 16 m away.

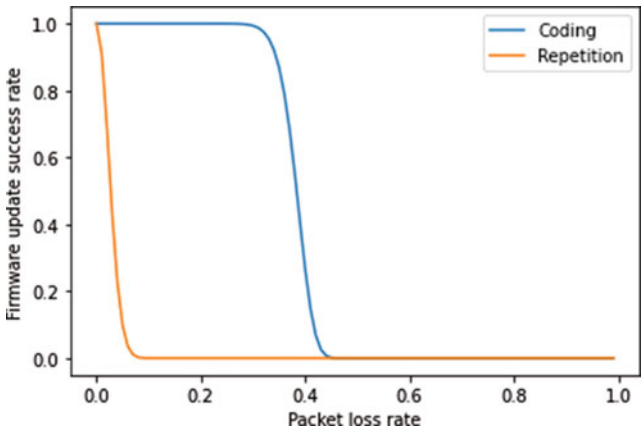

Fig. 50.3 Comparison of the probabilities of success using coding versus repetition

50.3.1 Operational Time

We set the physical link at a data rate of 250 kb/s. Each packet of a maximum length of 128 bytes (header + payload) has a transmission time about $128 * 8/250 = 4.09$ ms. Each page of the firmware contains $K = 37$ packets. In the coding protocol, these packets are encoded into $N = 74$ coded packets. In the repetition protocol, these $K = 37$ packets are transmitted twice. Regardless of the approach, 74 packets are transmitted for a page. We have added a sleep time of 0.1 ms after the transmission of each packet, and also a sleep time of 1 ms after each page. Recall that we assume a firmware has 25 pages. Thus, the total transmission time of a firmware is approximately

$$25 \cdot (74 \cdot (4.09 + 0.1) + 1) \text{ ms} = 7.78 \text{ s}. \tag{50.8}$$

In the experiment, we repeat the firmware transmission 100 times and wait 10 s after each transmission of entire firmware before starting the next one. It takes about 30 min to complete 100 firmware updates for each approach.

50.3.2 Experimental Results

In the settings shown in Fig. 50.4, a transmitter node broadcasts firmware to three receiver nodes placed at distances of 5 m away, 11 m away, and 16 m away. At each distance, the transmitter broadcasts 100 firmwares with each protocol, alternating between the repetition protocol and the coding protocol. Table 50.1 shows the packet loss rate at the receiver nodes as well as the firmware transmission success rates of two protocols at different distances.

It can be observed that as the distance between the transmitter and receiver increases, so does the packet loss rate. At a distance of 5 m, nearly all the packets are received correctly. At 11 m, the packet loss rate has increased to nearly 5%.

Fig. 50.4 Picture of the experimental setting

At the distance of 16 m, the packet loss rate has risen to about 27%. At longer distances, the performance differences between two protocols become apparent. According to the theoretical results shown in Fig. 50.3, we expect the success rate of the repetition protocol to drop off significantly, even with a very low packet loss rate. On the other hand, we expect the encoded protocol to be successful until the packet loss rate reaches around 40%. This is reflected in the experimental

Table 50.1 Experimental results

Receiver	RX 1	RX 2	RX 3
Distance (m)	5	11	16
Packet loss rate	0.54%	4.90%	27.20%
Repetition protocol success rate	100%	26%	0%
Coding protocol success rate	100%	100%	99%

results as well, with the encoded protocol outperforming the repetition protocol by far. Both protocols performed perfectly at a distance of 5 m, but at 11 m, the repetition protocol's success rate dropped to 26%. In contrast, the encoded protocol was still able to completely receive every firmware. At a distance of 16m, the difference is between the two protocols are even more apparent. The repetition protocol had virtually no chance for success, while the encoded protocol still successfully received 99% of the updates.

50.4 Conclusions

In this project, we proposed an alternative approach to wirelessly updating firmwares via repeated transmission by introducing erasure correction coding. We implemented the new protocol using TI LAUNCHXL-CC2652 LaunchPads. We tested the two protocols against each other in the environment of the ITE building at the University of Connecticut. According to our experimental data, the coding approach is much more efficient and reliable.

In this paper, we only encode packets within a page. In the future, we will improve the reliability further by applying the erasure correction coding at the page level. Furthermore, we will adapt the transmission protocol in the actual firmware update system, which consists of other modules like security subsystem and firmware update procedures on receiver nodes.

Acknowledgments We thank Ms. Sara Wrotniak for outlining the erasure correction coding approach in a course project report [5]. We thank Dr. Shengli Zhou, Dr. Song Han, and Mr. Zelin Yun for insightful discussions.

Funding B. Kivilcim was supported by the NSF grant (REU site) 1950600.

References

1. Texas Instruments, Firmware update process. Manual on https://software-dl.ti.com
2. K. Kerliu, A. Ross, G. Tao, Z. Yun, Z. Shi, S. Han, S. Zhou, Secure over-the-air firmware updates for sensor networks, in *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW)* (2019), pp. 97–100
3. S. Lin, D.J. Costello, *Error Control Coding*, 2nd edn. (Prentice Hall, 2004)
4. Texas Instruments, SimpleLink CC2640R2 Proprietary RF User's Guide, 2.70.0. Manual on https://dev.ti.com
5. S. Wrotniak, Improving Wireless Firmware Update Protocol for IoT Devices, final report of a course project, December (2021)