Protocol for Applying an Integrity-Assurance System Using Shortened Codes

Sydorenko Zoia Mykhailivna
Kharkiv National University of Radio Electronics, 14 Nauky Ave, Kharkiv UA-61166, Ukraine, zoia.sydorenko@nure.ua

Sievierinov Oleksandr Vasylovych
Kharkiv National University of Radio Electronics, 14 Nauky Ave, Kharkiv UA-61166, Ukraine, oleksandr.sievierinov@nure.ua

Abstract. *The paper examines a protocol for implementing a data integrity assurance system for the Industrial Internet of Things (IIoT) based on shortened Goppa codes. The main integrity threats in IIoT are analyzed, and it is shown that the presence of a cryptographic algorithm alone, without a proper protocol, does not guarantee the secure operation of the system. The use of a block-chaining protocol with the assignment of unique identifiers (UIDs) and timestamps is proposed to detect message replays and delays. A block-chaining encoding mechanism is described, which significantly reduces the likelihood that an attacker can form a valid codeword with a required UID. It is demonstrated that such a protocol minimizes risks associated with the linearity of shortened codes and provides an additional layer of protection, provided the Goppa polynomial key remains undisclosed..*

Keywords: *IIoT, data integrity, shortened Goppa codes, error-resilient coding, transmission protocol, unique identifier (UID), timestamps, block chaining.*

I. Introduction and Problem statement

The Industrial Internet of Things (IIoT) is a modern technology that integrates intelligent sensors, devices, and software to enable automated monitoring, data collection, and control of manufacturing processes. Despite significant benefits, IIoT deployment brings a number of serious security challenges that may lead to leakage of confidential information, financial losses, or physical damage to equipment and personnel. One of the primary challenges in IIoT security is ensuring information integrity – i.e., preserving data accuracy, authenticity, and immutability throughout its lifecycle, from collection to processing and storage [1, 2].

A breach of data integrity in IIoT systems can cause equipment malfunction, production incidents, and a significant loss of trust in the system. In industrial environments, where automated technologies and large numbers of interacting devices are used extensively, this problem becomes particularly acute.

Even small data distortions in industrial environments can have serious consequences. Data transmitted from sensors or consumed by automated systems must remain unchanged during transmission and storage. However, because of the large number of nodes and weak protection on some devices, this information can be intercepted, altered, or destroyed without detection.

To ensure data integrity, it is proposed to apply new approaches that combine modern cryptographic mechanisms and error-resilient coding [3].

The existence of a robust data integrity (authentication) algorithm alone does not guarantee that the tasks assigned to it will be solved. A cryptographic algorithm must be used within a set of rules or procedures – called a protocol – that provides the data integrity (authentication) required by the system [4]. Therefore, when designing a protection system it is necessary to solve two tasks: develop a resilient algorithm and a reliable protocol.

II. Problem solution and Results

An adversary attempting to disrupt the transmission system can use various actions: insert forged data, delay transmitted messages, disturb their order of delivery, or retransmit intercepted data.

A data-transfer protocol often includes an acknowledgment procedure: a transmitted packet is retransmitted if no positive acknowledgment is received. Retransmission repeated until a positive acknowledgment is obtained guarantees delivery of the message to the receiver. However, whenever a positive acknowledgment is lost, retransmission will result in a duplicate arriving at the receiver. The protocol must provide means to distinguish duplicates from primary packets.

Because of the linearity of the error-resilient coding used in the proposed method for ensuring data integrity, an attacker might attempt to construct a false codeword from intercepted messages.

Thus, the protocol under development must address the following tasks:

– protection against deletion (loss) or insertion of messages;
– protection against message delay;
– protection against disruption of message ordering;
– protection against duplication and replay of messages;
– protection against the possibility of forming a forged message based on intercepted transmitted data.

The protection mechanism against deletion (loss), insertion, ordering disruption, replays and duplication of messages consists in the transmitter appending a unique identifier (UID) to each message. Both transmitting and receiving modules must track and store the UID of the current packet. Each received packet is checked for errors. If an uncorrectable error configuration is present, the packet is rejected as corrupted and not processed.

If the packet is not corrupted or its errors are corrected, its UID is compared with the UID stored by the receiver. If the UIDs do not match, the message is discarded as forged. If they match, an acknowledgment of receipt is sent and the UID is advanced to the next value.

No duplicate packet – whether formed by the sender or by an adversary – will be accepted because its UID will not match the UID stored at the receiver. Any information packet lost during transmission will not be acknowledged and therefore will be retransmitted. A lost acknowledgment will cause the sender to retransmit the data packet and the receiver to resend its acknowledgment. If a packet arrives at the receiver before its predecessors, the UID check will cause such a packet to be discarded. Only the next-in-sequence packet may be accepted by the receiver for processing.

In most protocols, timestamps are used to detect adversary-induced message delays [5]. Like UIDs, timestamps also allow detection of replayed messages by an attacker.

Shortening Goppa codes used to construct the data-integrity system can lead to partially non-linear codes – that is, the sum of two codewords for such codes may not be a valid codeword. However, the linear combination of several codewords (more than two) may result in a valid codeword.

Thus, an adversary analyzing the data-transfer system can, with high probability, form a codeword that will be decoded and may even have the required UID. Although this forged codeword is most likely to decode to a bit sequence that is unreadable, it will be accepted for processing and may harm the system. A further drawback of a protection system using shortened Goppa codes is that two identical plaintext blocks produce identical codewords.

To mitigate these shortcomings it is proposed to use a protection system based on shortened Goppa codes together with a block-chaining protocol, shown in Fig. 1.

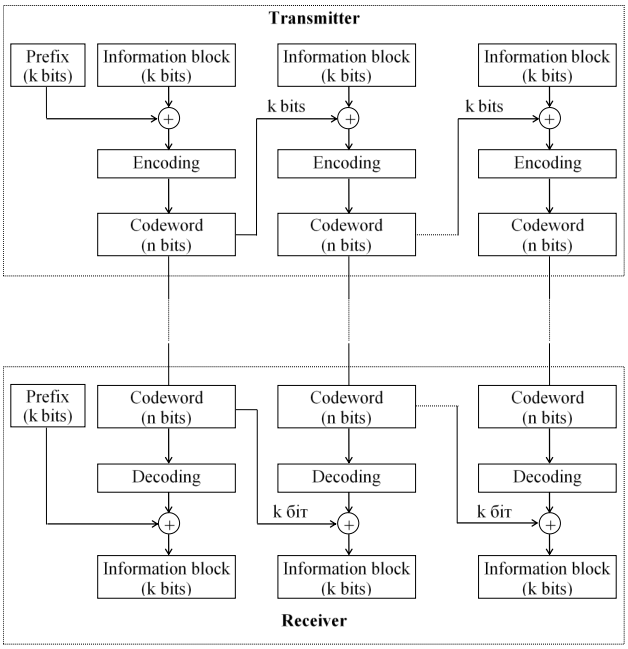

Figure 1. Block-chaining encoding protocol

Under this protocol, an information message is split into parts of k bits. The first information block is summed with a prefix block (this operation may be omitted); a timestamp can be used as such a prefix. The resulting block is sent to the encoder, whose output codeword is transmitted through the IIoT data channel. The first k bits of the transmitted codeword are then summed with the second information block and fed to the encoder. This process repeats until the entire message is encoded. On the receiving side, operations are performed in reverse order.

When using the block-chaining protocol, identical information messages yield identical codewords only when their prefixes are identical. This is not a significant problem because the prefix typically contains a changing timestamp value.

III. Conclusions

Therefore, when a protection system based on shortened Goppa codes is used together with a block-chaining protocol, an adversary will not be able to form, from intercepted data, a codeword with the required UID without revealing the Goppa polynomial key.

REFERENCES

[1]. Serror, M., Hack, S., Henze, M., Schuba, M., & Wehrle, K. (2020). Challenges and opportunities in securing the industrial internet of things. IEEE Transactions on Industrial Informatics, 17(5), 2985-2996.

[2]. Gebremichael, T., Ledwaba, L. P., Eldefrawy, M. H., Hancke, G. P., Pereira, N., Gidlund, M., & Akerberg, J. (2020). Security and privacy in the industrial internet of things: Current standards and future challenges. IEEE Access, 8, 152351-152366.

[3]. Yevheniev, A., Sydorenko, Z., & Sievierinov, O. (2025). Ensuring data integrity in industrial Internet of Things systems using error-correcting codes. Radiotekhnika, (221), 46–50. https://doi.org/10.30837/rt.2025.2.221.06

[4]. Blessing, A. I., Blessing, A., & Olawale, L. (2025). Strategies for ensuring data integrity and authenticity during the sharing process.

[5]. Ferrari, P., Flammini, A., Sisinni, E., Rinaldi, S., Brandão, D., & Rocha, M. S. (2018). Delay estimation of industrial IoT applications based on messaging protocols. IEEE Transactions on Instrumentation and Measurement, 67(9), 2188-2199.