Hindawi
Security and Communication Networks
Volume 2022, Article ID 8026280, 9 pages
https://doi.org/10.1155/2022/8026280

Research Article

IMCSA: Providing Better Sequence Alignment Space for Industrial Control Protocol Reverse Engineering

Yukai Ji,[1] **Tao Huang**,[1] **Chunlai Ma**,[2] **Chao Hu**,[3] **Zhanfeng Wang**,[4] **and Anmin Fu**[1]

[1]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[2]*National University of Defense Technology, Hefei, China*
[3]*PLA Army Engineering University, Nanjing, China*
[4]*Southeast University, Nanjing, China*

Correspondence should be addressed to Anmin Fu; fuam@njust.edu.cn

Received 9 September 2022; Revised 5 October 2022; Accepted 2 November 2022; Published 24 November 2022

Academic Editor: Naghmeh Moradpoor

Nowadays, with the wide application of industrial control facilities, industrial control protocol reverse engineering has significant security implications. The reverse method of industrial protocol based on sequence alignment is the current mainstream method because of its high accuracy. However, this method will incur a huge time overhead due to unnecessary alignments during the sequence alignment process. In this paper, we optimize the traditional sequence alignment method by combining the characteristics of industrial control protocols. We improve the frequent sequence mining algorithm, Apriori, to propose a more efficient Bag-of-Words generation algorithm for finding keywords. Then, we precluster the messages based on the generated Bag-of-Words to improve the similarity of the message within a cluster. Finally, we propose an industrial control protocol message preclustering model for sequence alignment, namely, IMCSA. We evaluate it over five industrial control protocols, and the results show that IMCSA can generate clusters with higher message similarity, which will greatly reduce the invalid alignments existing in the sequence alignment stage and ultimately improve the overall efficiency.

1. Introduction

The industrial control protocol is a collection of rules, standards, or conventions established for the exchange of data in the industrial control network, and it is an indispensable part of communication between entities of the Industrial Internet [1–3]. Therefore, industrial control protocols have become a common target for attackers to carry out network attacks. Conversely, industrial control protocols have also become the focus of defense against these attacks [4–6]. However, because of commercial and security factors, most industrial control protocol specifications are not open to the public [7, 8]. These private industrial control protocols pose a challenge for protocol analysis. As a result, industrial control protocol reverse engineering has gradually attracted the attention of industry and academia [9].

Industrial control protocol reverse engineering (ICPRE) is currently one of the most effective means to analyze unknown industrial control protocols, which can restore the protocol specification (protocol format, semantic information, and state transition information) by reverse analysis of the target protocol messages [10–12] without the target protocol design specification. The ICPRE method based on sequence alignment has become the most practical mainstream means at present because of its high accuracy. However, there are still some problems with the reverse analysis of industrial control protocols based on sequence alignment, especially in terms of time overhead. However, most existing related works[13–20] focus on how to introduce additional information to improve the accuracy of the format extraction results, while ignoring the potentially significant overhead. Therefore, we conducted a study to improve the efficiency of the algorithm.

The reverse approach based on sequence alignment utilizes a multiple sequence alignment algorithm (MSA) [21] to align sequences and calculate the similarity scores

between sequences. However, this method takes a lot of time for aligning message pairs. In particular, aligning two sequences with very low similarity is completely unnecessary because their alignment results will be discarded. To reduce these unnecessary sequence alignments, we provide a better alignment space for it by preclustering. The messages with high similarity will be clustered together and then sequence alignment will be performed in each cluster, which can effectively reduce unnecessary alignments and improve the efficiency of the whole algorithm.

1.1. Our Work. In this paper, we propose a Bag-of-Words generation algorithm for finding keywords of unknown protocol messages. Then, we apply it to a message clustering model, called IMCSA (industrial control protocol message preclustering for sequence alignment). The model can effectively enhance the accuracy of clustering results in the data preprocessing stage, thus reducing the time overhead of the sequence alignment process.

In summary, the contributions of this paper are as follows:

(1) We propose the Bag-of-Words generation algorithm (BWG) to find the frequent sequences by optimizing Apriori and we propose a frequent sequence discrimination strategy. We label the frequent sequences according to their frequencies, and those sequences that satisfy the threshold condition will be considered keywords. Furthermore, these labels can be used for feature extraction, and we can also align messages with the same format but different field values based on these tags in future work.

(2) We propose a message clustering model called IMCSA, which can cluster messages with high similarity without posterior probability. IMCSA provides a better alignment space for sequence alignment.

(3) We evaluate IMCSA over five industrial control protocols. Our experimental results show that IMCSA enables better clustering of industrial control protocol messages.

2. Related Work

In this section, we will discuss some current work related to our research.

2.1. Industrial Control Protocol Reverse Methods. To date, there is still relatively little work related to ICPRE, and the methods based on sequence alignment of network trace are still the currently dominant approaches. In 2004, Beddoe [22, 23] first introduced the biological alignment method into the protocol reverse work, and then he proposed the protocol informatics (PI) algorithm by combining Needleman&Wunsch and UPGMA. The sequence alignment algorithm has been reused in many subsequent works. In 2007, Cui et al. [14] improved the PI algorithm and introduced the concept of delimiters in a pioneering way. They applied it to Discover, which yielded good utility in analyzing textual protocols but did not work well for binary protocols.

Apart from that, some researchers have introduced heuristics to ICPRE. In 2018, Kai et al. [24] achieved accurate field segmentation and structure identification of industrial control protocols by combining taint analysis. However, it is difficult to implement and analyze such methods because of their high impact on the stability of the industrial control system and high controllability requirements. In 2020, Wang et al. [17] proposed an XGBoost model to achieve the extraction of industrial control protocol feature fields based on V-gram. However, the method is sensitive to the number and diversity of captured messages. In 2020, Zhao and Liu [13] applied deep learning techniques to protocol inversion and introduced temporality. The method has high generality and accuracy, but it requires a large number of samples to train the model. In 2021, Liu et al. [25] introduced the concept of entry distance and proposed IPRFW, which effectively locates the boundaries of fields using probabilistic methods. However, the method may filter out variable fields and is not robust.

2.2. Keyword Extraction Methods. At present, there have been several research works on keyword extraction. In 2013, Luo and Yu [26] improved the Apriori algorithm to extract frequent sequences of binary protocols as keywords for the problem of inverse analysis of binary protocols. The accuracy and efficiency of the sequence alignment algorithm were effectively improved, but the time complexity problem of the Apriori algorithm was not solved. In 2014, Bossert et al. [16] introduced contextual semantic information into the sequence alignment process and implemented Netzob. Netzob is still the mainstream analysis tool in the field of protocol reverse engineering. However, this method only introduces more information at the sequence alignment level to improve the accuracy but does not optimize its overhead. In 2014, Ji et al. [27] proposed a multi-mode matching algorithm for UAV protocol to extract keywords, and then Shim et al. [12] introduced message transmission direction to extract the features of Modbus. In 2021, Choi et al. [28] proposed a reverse method for the cryptographic protocol CAN. However, these works are not universal, and they are only applicable to certain specific protocols. In 2021, Wang et al. [20] introduced semantic information into the protocol reverse work, which greatly improved the accuracy of analysis results. However, the approach introduces a significant time overhead in the semantic premining phase.

In particular, Netplier [19] is the most representative scheme in recent years. In 2021, Ye et al. [19] proposed Netplier for the keyword extraction problem. By introducing multiple constraint relations and posterior probabilities, Netplier selects the candidate with the highest probability as the keyword. This method largely improves the accuracy of keyword searches and the correctness of clustering results. However, it introduces a very high overhead in the construction of the keyword candidate set and the judging of the candidates.

In summary, industrial control protocol reverse engineering is still in starting stage, and the sequence alignment-based methods are still the mainstream of industrial control protocol reverse work. However, most of the existing related research is focused on the accuracy of the reverse results in depth. The reverse method based on the sequence alignment algorithm has a high time overhead and there is still little work available on this problem.

3. Problem Definition

In this section, we first introduce the basic process of industrial control protocol reverse engineering. Then, we present the research focus and our system's design goals for this work.

3.1. ICPRE Model. As shown in Figure 1, the industrial protocol reverse work is divided into four main parts.

(i) *Data Preprocessing*. The data preprocessing phase formats the captured traffic to provide analysis samples for the protocol reverse work.

(ii) *Format Inference*. The format inference phase is the main part of the protocol reverse work, which is responsible for parsing the protocol format fields.

(iii) *Semantic Inference*. The semantic inference stage obtains the functional and semantic information of the field by analyzing the parsed field format.

(iv) *State Machine Inference*. The state machine inference phase constructs and simplifies the state machine to obtain protocol state transfer information.

The ultimate goal of our study is to reduce the time overhead of the sequence alignment algorithm in the format inference phase. In this paper, we focus our research on the data preprocessing stage. We optimize the data processing phase to provide an optimal sequence alignment space for the subsequent format inference. This will effectively avoid unnecessary alignments between low-similarity sequences and improve the overall efficiency.

Firstly, we define keywords as the words by which we can better cluster the messages together in their original symbol format. Then, we cluster the messages based on the keywords to improve the similarity of messages in the same cluster. With this approach, we can then reduce the invalid alignments in the format inference stage and improve the efficiency of the overall work.

3.2. Design Goals. We propose the design goals of IMCSA to address the problems. We believe that IMCSA should achieve the following goals:

(i) *Accuracy*. The accuracy of the clustering results is the primary goal of our proposed method. We hope that IMCSA will cluster messages of the same format without posterior probabilities.

(ii) *Efficiency*. We hope that the time overhead of the clustering phase will not affect the total overhead of the overall protocol reverse work.

(iii) *Applicability*. We hope that the proposed method and system can be applied to most industrial control protocols, not just to a few specific ones.

4. BW Generation Algorithm

In this section, we present the design details of our algorithm. We first introduce the basic algorithm of BWG. Then, we detail the functions of the various modules of the algorithm and core strategies.

4.1. Heuristic. By analyzing the design specifications of most public industrial control protocols, we propose two types of protocol characteristics: inter and intra-message characteristics [7, 19, 29].

Inter-message characteristics refer to the features and patterns exhibited between different messages during the communication process. Generally speaking, the number of protocol formats is limited because of the simplicity and practicality of the protocol design. The formats in the two directions of request and response are also different. Usually, there is an extra field (e.g., confirmation message, error code, and so on) in the format of the response. In addition, the symbols of each complete communication process usually exhibit periodic characteristics.

Intra-message features refer to the features displayed between different fields in a message, which is also the starting point for most of the current work. Due to the readability of the protocol design, the number of fields used by a protocol is limited. Moreover, each field contained in each message has its meaning, and the field information except the carried data information constitutes the dictionary of the protocol specification. In addition, there are also some relationships between the fields before and after each message (for example, the value of the length field represents the length of the field it points to).

Previous work [14] has expressed the fields as TLV (type-length-value). Also, the type is based on the statistical characteristics represented by its value. In this paper, according to the frequency of the sequence in the message, we label the type as C (fixed value), H (high-frequency value), and L (low-frequency value), respectively.

4.2. Basic Algorithm. Luo and Yu [26] used the Apriori algorithm to mine frequent sequences. They selected two sequences with the highest frequency at a time to merge and use the merged sequence as a candidate to search again. This method has a high time overhead. Based on the basic idea of the Apriori algorithm, we draw the following conclusions:

(i) The occurrence frequency of a sequence is always greater than or equal to its superstring and less than or equal to its substring.

(ii) If the subsequence is not a frequent string, then the superstring must also not be a frequent string.

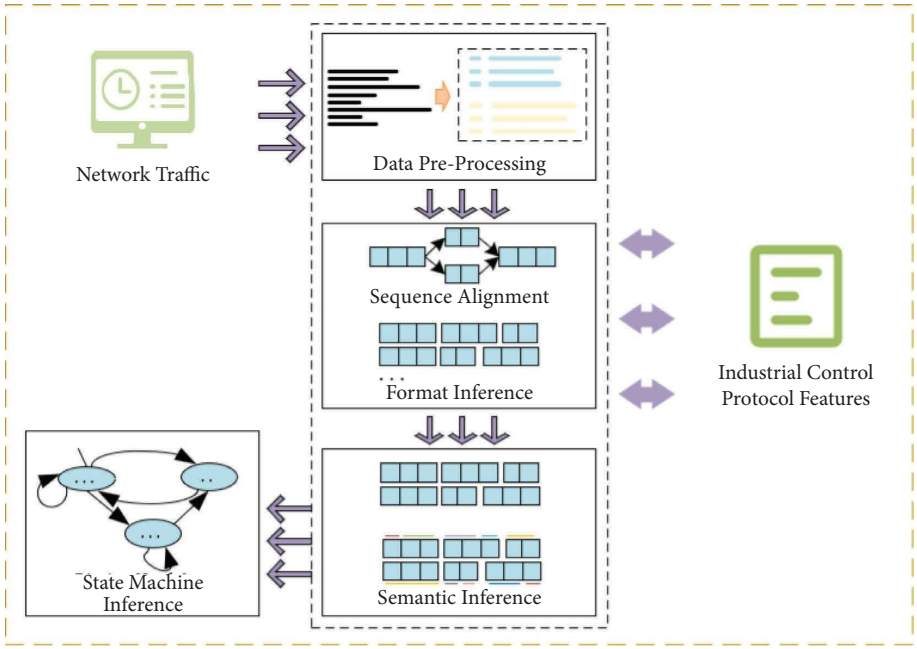

Figure 1: The model of industrial control protocol reverse engineering.

Likewise, if the superstring is a frequent sequence, then the substring must also be a frequent sequence.

On this basis, we have improved the Apriori algorithm. We call the improved algorithm the Bags-of-Words Generate (BWG). We obtain the new string by expanding the old string on the message sequence. Then, we directly determine the relationship between the new string and the original string. Finally, we operate on it according to different determination strategies. The algorithm of BWG is shown in Algorithm 1.

The algorithm consists of four modules, Initialize, Expand, Judge, and Vote.

Initialize. In this module, we use the current traversal character as the initialization sequence. Also, we calculate the key threshold information required during the loop based on the message information.

Voting. In this module, we compute the frequency of the input sequence using a voting algorithm. For the input candidate s, all voters mi in the message set M vote for it. If the voter m_i contains s, then vote for it. Eventually, the vote percentage of s is obtained based on the voting, and the vote percentage is used as the frequency of that sequence in the message set as the output.

Expand. In this module, we expand the old sequence backward one character at a time (half a byte in hexadecimal messages) according to the offset position of the old sequence in the message m_i. We take the dilated sequence as output.

Judge. This module is the core of this algorithm. In this module, we take the old and new sequences and their vote percentage as input. Also, the output is a keyword discrimination strategy. Typically, the two fields in a protocol message exhibit different entropy values at their splits [30]. This is because the message is a combination of fields, and the arrangement and combination of the fields determine the format and semantics of the packet. When the character occurrence frequency entropy value changes drastically, we can consider that the fields are boundary values. Therefore, based on the above ideas, we make different determinations for the input. The determination strategies are shown in Algorithm 2.

5. IMCSA

In this section, we present the details of the proposed system. We first introduce the system architecture. Then, we detail the various modules of IMCSA. Finally, we present the calculation of the significant threshold.

5.1. System Architecture. As shown in Figure 2, IMCSA is composed of three main modules:

(1) *Message Preprocessing Module*. In this module, based on the multi-layer structure of the protocol design, we extract the known information of the underlying traditional protocol and the data information of the unknown protocol of the target layer. Then, we preprocess the message based on the extracted information.

(2) *BW Generation Module*. In this module, we use the proposed BWG to process the target layer protocol information. Then, we obtain the keyword set B.

(3) *Feature Extraction and Clustering Module*. In this module, we perform feature extraction for messages

Input: preprocessed message set: M; threshold: τ
Output: Bag-of-Word set B

for $m_i \in M$ **do**
  old_str← initialize.
  forHalf_Byte$_j \in m_i$**do**
    old_fre = cal_fre(old_str);
    ifold_fre > = τ**then**
      new_str = Expand(old_str);
      new_fre = cal_fre(new_str);
      ifnew_fre > τ**then**
        Judge(old_str, new_str, B);
      end
      else
        old_str ⟶ B;
        old_str = new_char;
      end
    end
  end
end
return B.

Algorithm 1: BW generation.

Input: string before and after expansion: old_str, new_str; expanded character: new_char; threshold: u; Bag-of-Word: B
Output: judgement: case

old_fre = cal_fre(old_str)
new_fre = cal_fre(new_str)
new_chars = expand(new_char)
chars_fre = cal_fre (new_chars)
if old_fre == 1 **then**
  ifnew_fre! = 1&& *len*(old_str > = 2)**then**
    label (old_str)
    case1: old_str ⟶ B; old_str = new_char
  end
  else
    case2: old_str = new_str
  end
end
if old_fre < 1 **then**
  ifchars_fre == 1**then**
    label (old_str)
    case1: old_str ⟶ B; old_str = new_char
  end
  else
    ifabs(new_fre − old_fre) > = u**then**
      label (old_str)
      case1: old_str ⟶ B; old_str = new_char
    end
    else
      case2: old_str = new_str
    end
  end
end
return case.

Algorithm 2: Keyword discrimination strategy.

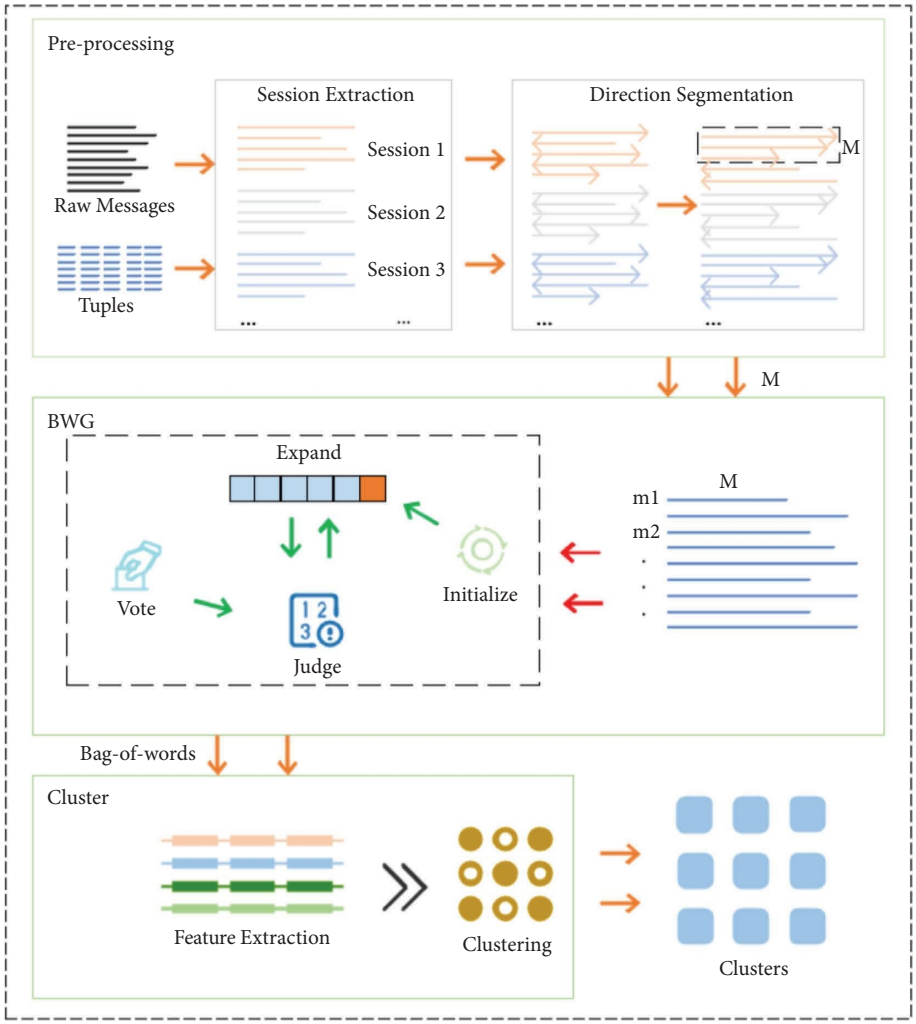

Figure 2: Architecture of IMCSA.

based on the frequent sequences in B. Finally, the messages are clustered by the feature vectors.

5.2. Message Preprocessing. In this module, we extract the five-tuple of messages, denoted as < timestamp, src_ip, dst_ip, src_port, dst_port >. Then, we process the message based on the five-tuple in the following steps.

(i) *Session Extraction*. The target of our analysis is the network traces, which is the captured message traffic. Typically, the unknown protocols we analyze are usually in the application layer. Based on other existing public protocol information, we can extract a lot of useful information. Then, we can reconstruct sessions based on them.

(ii) *Direction Segmentation*. Protocols are designed to enable the communication of data between entities. In general, the complete communication process consists of two phases: connection establishment and data transmission, and the protocol formats display different characteristics in these two phases. Therefore, we divide the reconstructed sessions into sets of different directions based on the extracted five-tuple information.

By processing the above steps, we divide the message into sets with the same session and in the same direction. The sequence of messages after the data preprocessing is represented as

$$M = \langle m_1, m_2, m_3, \ldots, m_n \rangle. \tag{1}$$

5.3. BW Generation. In this module, we use the BWG algorithm described above to process the sequence of preprocessed messages. With the BW algorithm, we can obtain a collection of keywords for the sample messages, called Bag-of-Words, which are represented as

$$B_i = \langle w_1 : t_1; w_2 : t_2; \ldots, w_n : t_n \rangle, \tag{2}$$

where w_i denotes the keywords and t_i denotes the tag information. For example, $B_i = \langle k1 : C; k2 : H; k3 : L \rangle$ indicates that $k1$ is fixed value, $k2$ is high-frequency value, and $k3$ is low-frequency value.

5.4. Feature Extraction and Clustering. To eliminate the effect of variable-length fields on the offsets, we introduce forward and backward offsets, where

$$\begin{cases} \text{offset\_forward}_k = \sum_{i=0}^{k-1} l_i, \\ \text{offset\_backward}_k = \sum_{i=k+1}^{n} l_i. \end{cases} \tag{3}$$

Then, we represent each message as a sequence of pattern strings S and generate the corresponding feature vector V, which is represented as

$$\begin{cases} S_i = t(l_1) - t(l_2) - t(l_3) - \cdots - t(l_n), \\ V_i = \langle (w_1 \colon l_1 \colon o\_f_1 \colon o\_b_1), \ldots, \{w_n \colon l_n \colon o\_f_n \colon o\_b_n\} \rangle, \end{cases} \tag{4}$$

where t_k denotes the tag of the field, l_k denotes the length of the field, and o_f_k and o_b_k denote the forward offset and backward offset, respectively.

Finally, we cluster messages based on the feature vectors.

5.5. Threshold Calculation. Finally, we will discuss the calculation of the important threshold in our algorithm.

Previous work in determining keywords usually compares the frequency of string occurrences with a predefined threshold. If the frequency is greater than the threshold, it is considered to be a frequent sequence. In these works, the threshold is usually set to a fixed value. However, the threshold that determines whether a string is a keyword or not should simply not be a fixed value. Due to the characteristics of the message and the string itself, other factors can affect this threshold, such as the length of the string. We find that the length of the string is inversely proportional to the threshold. As an extreme example, when the candidate string is very short, it cannot be directly defined as a keyword even if it occurs very frequently. Conversely, when the candidate keyword is very long, we can be lenient with it. Given a candidate string s, the frequent threshold $\tau(s)$ of the candidate string is obtained by the following equation:

$$\tau(s) = c \times e^{-(\text{len}(s)/5)}, \tag{5}$$

where c is a constant and relates to the intended target and $\text{len}(s)$ denotes the length of string s.

6. Implementation and Evaluation

In this section, we first introduce the implementation of the IMCSA. Then, we evaluate its performance through comparative experiments.

6.1. Experiment Setup

6.1.1. Implementation. We have implemented IMCSA using Python and deployed it on a Windows system, and our experiment was carried out on AMD R7-5800H, 3.2 GHz, with 16 GB memory.

The goal of IMCSA is to improve the accuracy of message clustering results. To evaluate the effectiveness of IMCSA, we have chosen Netzob [16] and AutoReEngine [26] as baselines. Among them, AutoReEngine does not release its source code, and we replicated it according to the method mentioned in the article [26]. Then, we performed cluster analysis on the dataset using each of the three tools.

6.1.2. Datasets. To test the applicability of IMCSA, we selected five industrial control protocols. Table 1 shows the statistical information of the dataset [31]. For each protocol, we filtered 300 messages for the experiment except Heart due to lack of enough messages. Moreover, we used Netplier [19] which introduced a posteriori probabilities to detect true keywords and obtain the true format, to extract the true format of the messages in the dataset.

6.1.3. Metrics. In this paper, we use two common metrics [32] for clustering performance evaluation to assess the accuracy of our clustering results.

(i) *Homogeneity Score*. Homogeneity implies whether the samples in each cluster of the clustering result are sufficiently similar; in this paper, it means whether messages in the same cluster belong to the same true format.

(ii) *Completeness Score*. Completeness means whether samples with high similarity are clustered into one cluster; in this paper, it means whether messages of the same format belong to the same set.

6.2. Experiment Result

6.2.1. Accuracy. As shown in Figures 3 and 4, IMCSA outperforms the other two tools clearly. As we have observed, Netzob and AutoReEngine make it hard to balance both homogeneity and completeness. Netzob generates more clusters to improve the similarity of messages within clusters; however, this also creates redundancy. This is because Netzob selects words with fixed offset positions as keywords for clustering, which leads to a low completeness score. On the contrary, AutoReEngine selects frequent sequences as keywords for clustering, which effectively reduces redundancy. However, it does not perform well in terms of accuracy because the frequent set is not accurate enough. IMCSA produces considerable improvements in both aspects through algorithm improvements. Moreover, since we have chosen five different protocols for experimentation in different environments, this reflects the applicability of our system.

6.2.2. Time Overhead. Furthermore, we conducted performance tests on IMCSA to evaluate its efficiency in finding keywords. As shown in Figure 5, IMCSA is more efficient than AutoReEngine in finding keywords. This is because we optimize the construction of frequent sequence candidate sets, which reduces the number of loops of the algorithm.

Table 1: Introduction of dataset.

Protocol	Message information		
	Data size (K)	Numbers of messages	Message types
dnp3	103	314	6
dhcp	51	300	4
modbus	40	300	8
Heart	2	24	3
smb	69	306	11

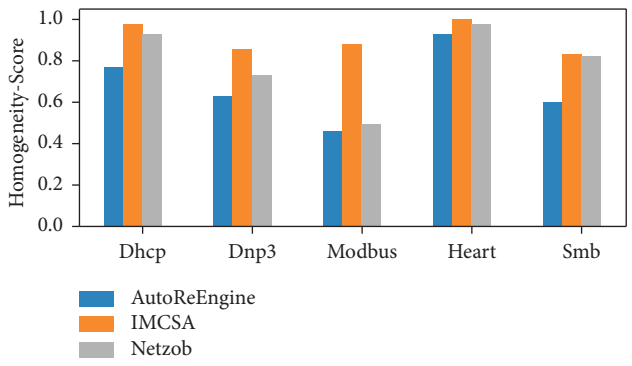

Figure 3: Homogeneity score of clustering result.

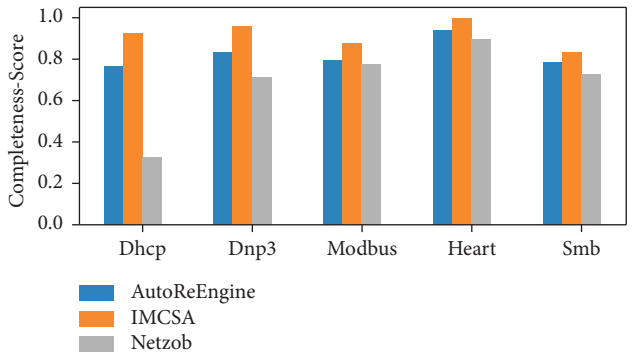

Figure 4: Completeness score of clustering result.

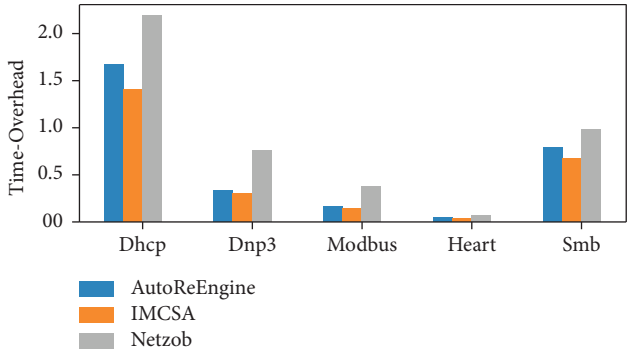

Figure 5: Time overhead of clustering.

Netzob's performance is slightly slower in terms of efficiency because it calculates the additional information needed for the subsequent stages.

In summary, IMCSA shows good performance in clustering industrial control protocol messages without additional time overhead, which improves the similarity of messages within clusters and effectively reduces redundancy.

7. Conclusion

In this paper, to address the problem of the high time overhead of existing reverse methods for industrial control protocols based on sequence alignment, we analytically transform the problem into how to provide a better sequence matching space. On this basis, we propose the BGW algorithm for finding keywords of protocol messages and implement the BGW-based IMCSA clustering model to provide a better sequence alignment space. IMCSA can improve the clustering effect in the message preprocessing stage and reduce unnecessary alignment in the sequence alignment stage. Our experimental evaluation shows that IMCSA possesses good applicability and performs well in industrial control protocol message clustering and can improve the efficiency of the algorithm without degrading the overall algorithm accuracy.

Future work will combine tag information to optimize sequence alignment and semantic inference stages.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (62072239), Future Network Scientific Research Fund Project (FNSRFP-2021-ZD-05), and Fundamental Research Funds for the Central Universities (30921013111 and 30920021129).

References

[1] A. Kim, J. Oh, K. Kwon, and K. Lee, "Consider the consequences: a risk assessment approach for industrial control systems," *Security and Communication Networks*, vol. 2022, Article ID 3455647, 19 pages, 2022.

[2] A. Fu, X. Zhang, N. Xiong, Y. Gao, H. Wang, and J. Zhang, "VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3316–3326, 2022.

[3] B. Kuang, A. Fu, Y. Gao, Y. Zhang, J. Zhou, and R. H. Deng, "FeS. A.: Automatic federated swarm attestation on dynamic

large-scale IoT devices," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[4] Z. Chen, A. Fu, R. H. Deng, X. Liu, Y. Yang, and Y. Zhang, "Secure and verifiable outsourced data dimension reduction on dynamic data," *Information Sciences*, vol. 573, no. 6, pp. 182–193, 2021.

[5] J. Yan, Z. Du, J. Li, S. Yang, J. Li, and J. Li, "A threat intelligence analysis method based on feature weighting and BERT-BiGRU for industrial Internet of things," *Security and Communication Networks*, vol. 2022, Article ID 7729456, 11 pages, 2022.

[6] H. Qiu, H. Ma, Z. Zhang et al., "RBNN: memory-efficient reconfigurable deep binary neural network with IP protection for Internet of things," 2021, https://arxiv.org/abs/2105.03822.

[7] Y. Huang, H. Shu, F. Kang, and Y. Guang, "Protocol reverse-engineering methods and tools: a survey," *Computer Communications*, vol. 182, pp. 238–254, 2022.

[8] C.-M. Chen, X. Li, S. Liu, M.-E. Wu, and S. Kumari, "Enhanced authentication protocol for the Internet of things environment," *Security and Communication Networks*, vol. 2022, Article ID 8543894, 13 pages, 2022.

[9] S. Kleber, L. Maile, and F. Kargl, "Survey of protocol reverse engineering algorithms: decomposition of tools for static traffic analysis," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 526–561, 2019.

[10] F. Meng, C. Zhang, and G. Wu, "Protocol reverse based on hierarchical clustering and probability alignment from network traces," in *Proceedings of the 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA)*, pp. 443–447, IEEE, Shanghai, China, March 2018.

[11] Y. H. Goo, K. S. Shim, M. S. Lee, and M. S. Kim, "Protocol specification extraction based on contiguous sequential pattern algorithm," *IEEE Access*, vol. 7, Article ID 36057, 2019.

[12] K. Shim, Y. Goo, M. Lee, and M. Kim, "Clustering method in protocol reverse engineering for industrial protocols," *International Journal of Network Management*, vol. 30, no. 6, 2020.

[13] R. Zhao and Z. Liu, "Analysis of private industrial control protocol format based on LSTM-FCN model," in *Proceedings of the 2020 International Conference on Aviation Safety and Information Technology*, pp. 330–335, Weihai, China, October 2020.

[14] W. Cui, J. Kannan, and H. J. Wang, "Discoverer: automatic protocol reverse engineering from network traces," in *Proceedings of the USENIX Security Symposium*, pp. 1–14, Boston, MA, USA, August 2007.

[15] J. Caballero, H. Yin, Z. Liang, and D. Song, "Polyglot: automatic extraction of protocol format using dynamic binary analysis," in *Proceedings of the ACM Conference on Computer & and Communications Security*, ACM, Alexandria, VA, USA, November 2007.

[16] G. Bossert, F. Guihéry, and G. Hiet, "Towards automated protocol reverse engineering using semantic information," in *Proceedings of the 9th ACM symposium on Information, computer, and communications security*, pp. 51–62, Kyoto Japan, June 2014.

[17] R. Wang, Y. Shi, and J. Ding, "Reverse engineering of industrial control protocol by XGBoost with V-gram," in *Proceedings of the 2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pp. 172–176, IEEE, Chengdu, China, December 2020.

[18] X. Wang, K. Lv, and B. Li, "IPART: an automatic protocol reverse engineering tool based on global voting expert for industrial protocols," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 35, no. 3, pp. 376–395, 2020.

[19] Y. Ye, Z. Zhang, F. Wang, X. Zhang, and D. Xu, "NetPlier: probabilistic network protocol reverse engineering from message traces," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2021, https://www.ndss-symposium.org/ndss-paper/netplier-probabilistic-network-protocol-reverse-engineering-from-message-traces/.

[20] Q. Wang, Z. Sun, Z. Wang et al., "A practical format and semantic reverse analysis approach for industrial control protocols," *Security and Communication Networks*, vol. 2021, Article ID 6690988, 11 pages, 2021.

[21] D.-F. Feng and R. F. Doolittle, "Progressive sequence alignment as a prerequisitetto correct phylogenetic trees," *Journal of Molecular Evolution*, vol. 25, no. 4, pp. 351–360, 1987.

[22] Phreakocious, "Protocol Informatics - Tools for Binary Protocol Analysis," http://phreakocious.net/PI/.

[23] M. A. Beddoe, *Network Protocol Analysis Using Bioinformatics Algorithms*, Toorcon, vol. 26, no. 6, , pp. 1095–1098, 2004.

[24] C. Kai, Z. Ning, W. Liming, and X. Zhen, "Automatic identification of industrial control network protocol field boundary using memory propagation tree," in *Proceedings of the International Conference on Information and Communications Security*, pp. 551–565, Springer, Cham, Wuhan, China, September 2018.

[25] O. Liu, B. Zheng, W. Sun et al., "A data-driven approach for reverse engineering electric power protocols," *Journal of Signal Processing Systems*, vol. 93, no. 7, pp. 769–777, 2021.

[26] J. Z. Luo and S. Z. Yu, "Position-based automatic reverse engineering of network protocols," *Journal of Network and Computer Applications*, vol. 36, no. 3, pp. 1070–1077, 2013.

[27] R. Ji, H. Li, and C. Tang, "Extracting keywords of UAVs wireless communication protocols based on association rules learning," in *Proceedings of the 2016 12th International Conference on Computational Intelligence and Security (CIS)*, pp. 309–313, IEEE, Wuxi, China, December 2016.

[28] W. Choi, S. Lee, K. Joo, H. J. Jo, and D. H. Lee, "An enhanced method for reverse engineering CAN data payload," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3371–3381, 2021.

[29] Y. Wang, X. Li, J. Meng, Y. Zhao, Z. Zhang, and L. Guo, "Biprominer: automatic mining of binary protocol features," in *Proceedings of the 2011 12th International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 179–184, IEEE, Gwangju, Republic of Korea, October 2011.

[30] Z. Zhang, Z. Zhang, P. P. C. Lee, Y. Liu, and G. Xie, "Toward unsupervised protocol feature word extraction," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 10, pp. 1894–1906, 2014.

[31] Github, "Modbus Trace," https://github.com/ITI/ICS-SecurityTools/blob/master/pcaps/bro/modbus/modbus.pcap.

[32] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLPCoNLL)*, pp. 410–420, Prague, Czech Republic, June 2007.