IEICE TRANS. INF. & SYST., VOL.E100–D, NO.8 AUGUST 2017

PAPER Special Section on Information and Communication System Security

Recovery Measure against Disabling Reassembly Attack to DNP3 Communication

Sungmoon KWON[†a)], **Hyunguk YOO**[†], ***Nonmembers***, ***and*** **Taeshik SHON**[††b)], ***Member***

SUMMARY In the past, the security of industrial control systems was guaranteed by their obscurity. However, as devices of industrial control systems became more varied and interaction between these devices became necessary, effective management systems for such networks emerged. This triggered the need for cyber-physical systems that connect industrial control system networks and external system networks. The standards for the protocols in industrial control systems explain security functions in detail, but many devices still use nonsecure communication because it is difficult to update existing equipment. Given this situation, a number of studies are being conducted to detect attacks against industrial control system protocols, but these studies consider only data payloads without considering the case that industrial control systems' availability is infringed owing to packet reassembly failures. Therefore, with regard to the DNP3 protocol, which is used widely in industrial control systems, this paper describes attacks that can result in packet reassembly failures, proposes a countermeasure, and tests the proposed countermeasure by conducting actual attacks and recoveries. The detection of a data payload should be conducted after ensuring the availability of an industrial control system by using this type of countermeasure.

key words: *industrial control system, DNP3, reassembly, availability*

1. Introduction

In the past, the security of industrial control system networks was guaranteed because these networks were not connected to external networks. However, current industrial control system networks interface with existing external networks through a cyber-physical system in order to manage devices located at many different sites. Although a cyber-physical system can enable the efficient control of many devices from a control center, access routes from external networks to industrial control system networks now exist. Cyber-attacks through these routes are a serious threat to old and nonsecure equipment. Thus, some protocols used in industrial control systems specify security functions in their standards, but security patching runs the risk of malfunctions and is difficult owing to the nature of industrial control systems. Furthermore, certain older equipment with low performance should be replaced to achieve better security.

Considering these circumstances, T. Mander *et al.* [1] conducted a rule-based study to detect the attacks of a payload by making payload rules, and Carcano *et al.* [2] covered the overall fields of the DNP3 protocol for whitelist rules to detect attacks. H. Lin *et al.* [3] conducted a study to detect attacks based on the specification information of the DNP3 protocol, and Yun *et al.* [4] conducted anomaly detection using burst-based communication patterns. J. Nivethan *et al.* [5] described a Linux-based firewall for the DNP3 protocol, and A. Shahsavari *et al.* [6] and H. Xu *et al.* [7] conducted data payload analyses of smart grids.

Unfortunately, all of these techniques focus on detecting attacks against data payloads and do not examine packet reassembly failures. This can be achieved by considering availability, which is the most important characteristic in the operation of an industrial control system. East *et al.* [8] described attacks that disrupted the reassembly of a message. The availability of the industrial control system must be ensured before detecting a data payload. This can be achieved by eliminating problems in obtaining data by recovering that data when the packet reassembly information is incorrect. After this procedure, the detection of data payloads should be conducted.

In this paper, we examine attacks that target Distributed Network Protocol 3.0 (DNP3) and disable the packet reassembly. We propose and test a countermeasure to these attacks. The DNP3 protocol is widely used in the Industrial Control Systems (ICS) network. More than 75% of power utilities in North America reported using the DNP3 protocol as part of their Supervisory Control and Data Acquisition (SCADA) system [9]. Therefore, we chose the DNP3 protocol to describe attacks and recovery measures. Note that these attacks and recovery measures can be applicable to other ICS protocols such as Modbus.

Our paper is structured as follows. We describe the DNP3 protocol and its architecture in Sect. 2. We present attack cases by fields containing the main information on reassembly, and propose recovery measures, in Sect. 3. In Sect. 4, we verify these measures by conducting actual attacks. We present the conclusion of this paper in Sect. 5.

2. DNP3 Protocol

In this section, we describe the DNP3 protocol, its current security status, and contents related to packet reassembly.

2.1 DNP3 Protocol and Its Security

DNP3 is a protocol that is widely used to obtain data and

Manuscript received September 8, 2016.
Manuscript revised February 5, 2017.
Manuscript publicized May 18, 2017.

[†]The authors are with Department of Computer Engineering, Ajou University, Suwon, South Korea.
[††]The author is with Division of Information and Computer Engineering, Ajou University, Suwon, South Korea.
a) E-mail: calmcombat@gmail.com
b) E-mail: tsshon@ajou.ac.kr (Corresponding author)

DOI: 10.1587/transinf.2016ICP0026

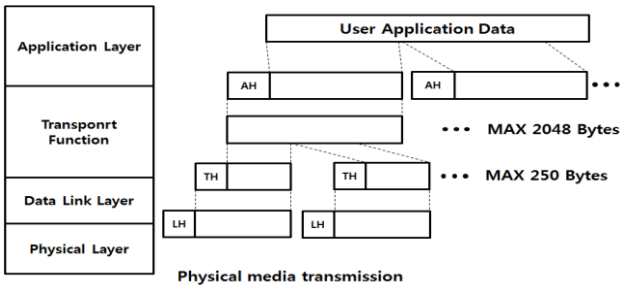

Fig. 1 Transport function architecture

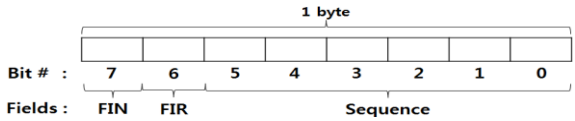

Fig. 2 Transport function architecture

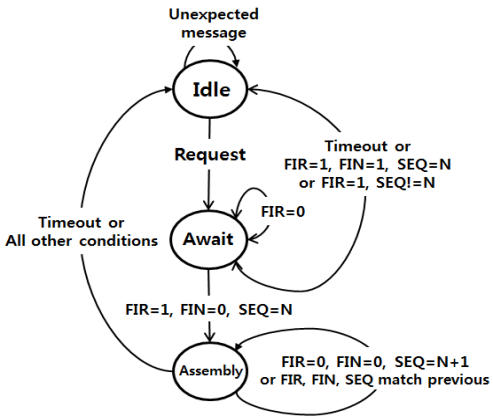

Fig. 3 State diagram according to FIN, FIR, and sequence

remote control in an industrial control system. DNP3 was developed in 1994 and was adopted as the IEEE 1815-2010 [10] standard in 2010. Its latest version is IEEE 1815-2012 [11]. A newer version is under development. DNP3 had been used for a long period since its development in 1994. There was research into security for the DNP3 protocol, including cryptographic technologies, before 2010 [12], [13]. Since then, DNP3 has provided Secure Authentication as a security capability.

However, Secure Authentication V2 of IEEE 1815-2010 has been deprecated, and only Secure Authentication V5 of IEEE 1815-2012 can provide security that is suitable for the current environment [15]. Therefore, older products cannot support Secure Authentication V5 or should be updated to support it. A number of industrial control systems still use the DNP3 protocol but do not support security functions. To detect attacks targeting the DNP3 protocol, many studies [1]–[5] have been conducted.

However, these studies focus on detecting attacks on reassembled packets, and do not address attacks on the fragments before reassembling. East *et al.* [8] showed the vulnerability of fragmented packets, and Lee *et al.* [14] conducted modification attacks to DNP3 that could be used to attack fragmented packets.

Therefore, when applying detection techniques [1]–[7], the availability is infringed. In addition, the detection engine could become useless because data payload information cannot be reassembled normally, even when the reassembly information of a DNP3 packet is wrong by only 1 bit. Thus, correct packet reassembly should be conducted before the detection of a data payload. This can also minimize any infringements on availability.

2.2 DNP3 Protocol Architecture and Its Treatment Method

The DNP3 protocol is largely divided into a data link layer, Transport Function, and application layer, as shown in Fig. 1. In the application layer, when the size of the data to be transmitted is too large to be treatable, the data becomes fragments with an Application Header (AH). In the transport function, a fragment that is larger than the transmittable size can again be divided into segments with a Transport Header (TH). Generally, however, data of a treatable size in the transport function is used. Without further segmentation, and with a Data Link Header (LH) added, the message is finally sent through the physical layer. This paper focuses on the Transport Function layer. As shown in Fig. 2, the Transport Function is only 1 byte. It is composed of FIN (Final) and FIR (First) of 1 bit and a Sequence of 6 bits. FIN and FIR indicate the beginning and end of a segment, respectively, and the Sequence is used to detect duplicate and missing segments.

A state diagram for a data acquisition message that includes FIN, FIR, and Sequence is shown in Fig. 3. This is defined in the IEEE 1815-2012 standard. If FIR, FIN, and Sequence are abnormal, that segment, already-received segments, and subsequent segments are discarded, and the state returns to idle (the state that exists before sending a request). Consequently, the master cannot acquire the data for the request. However, when a newly-received segment's sequence is the same as the previous sequence, only the newly-received segment is discarded, and the master waits for the next normal segment.

Except for data acquisition, the control command is same in that it always discards the message. However, because the control command directly affects the equipment, even a 1-bit error cannot be undervalued. Therefore, when the sequence of a control command is abnormal, someone only follows command discarding procedure after discarding segment it as shown in standard, it is not an object of reassembly recovery dealt in this paper.

3. Attack on DNP3 Reassembly Cases, and Recovery Measure

In this section, we describe possible attacks and a recovery measure with regard to packet reassembly.

Since DNP3 does not specify the names of data points,

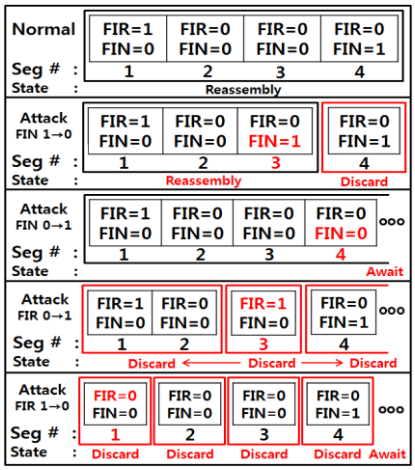

Fig. 4 Manipulation of FIN and FIR

it is impossible to know what data is indicated by each data point without having additional information. Thus, when an attacker manipulates data arbitrarily without prior knowledge of a data point, the attack fails to achieve meaningful results. By contrast, even a 1-bit change in the field of the Transport Function can make the reassembly of a packet impossible, and the packet will be discarded. This can seriously affect the availability. Depending on the implementation, incorrect changes to the Cyclic Redundancy Check (CRC) could also produce the same results.

To perform an attack, the attacker must be able to sniff network packets and transmit manipulated FIN/FIR/Sequence packets. An attacker with these capabilities can infringe on the availability of a control system by making a change of only 1 bit to a FIN/FIR/Sequence field.

3.1 Manipulation of FIN and FIR

DNP3 indicates the beginning and the end of a segment as FIR and FIN, respectively, and does not have a field that defines the length of a message before it is segmented. Therefore, when even 1 bit of information from a FIN or FIR is transmitted abnormally, correct reassembly is impossible. For example, if an attacker manipulates FIN of a segment from 0 to 1, the segment is recognized as the final segment, and the subsequent segment is discarded. By contrast, when an attacker manipulates FIN from 1 to 0, the system will wait for a subsequent segment.

If an attacker manipulates FIR of a segment from 0 to 1, the system will recognize it as the beginning of a new response, and because a new response starts without receiving a segment with an FIN of 1, the system will discard the already-received segment, the manipulated segment, and the subsequent segment. By contrast, when an attacker manipulates FIR of a segment from 1 to 0, the system will discard the segment and wait for a segment in which FIR is 1, and will wait until a time-out occurs. An example of the manipulation of FIN and FIR is shown in Fig. 4.

No.	Protocol	Info
6916	DNP 3.0	from 1 to 8 , Read, Internal Indications
6917	DNP 3.0	from 8 to 1 , Unsolicited Response
6918	DNP 3.0	from 1 to 8 , len=5, ACK
6919	DNP 3.0	from 8 to 1 , len=5, ACK
6920	DNP 3.0	from 1 to 8 , Confirm
6921	DNP 3.0	from 8 to 1 , len=5, ACK
6922	DNP 3.0	from 8 to 1 , Response
6923	DNP 3.0	from 1 to 8 , len=5, ACK

Fig. 5 Unsolicited response between read request and solicited response-captured packet using Wireshark

3.2 Sequence Manipulation

The DNP3 protocol recommends that when you transmit packets by fragmenting, you should transmit the fragments one by one, rather than one packet at a time, and receive an ACK packet in return. In addition, even when using User Datagram Protocol (UDP), the fragments can be out of order because they are all transmitted at one time. Actually, when we analyzed DNP3 protocol packets from energy control systems, there were no packets that were out of order. Therefore, manipulating a sequence in general situations may not be a significant issue.

However, sequence manipulation could be a problem if the master receives (in addition to a solicited response for a request) many responses, including unsolicited responses that are generated by the equipment without a request message. This is shown in Fig. 5, where there is a read request from the master (6916), and the master receives an unsolicited response from outstation (6917). Then, the master receives a solicited response (6922). In this situation, the master needs to determine whether the received response is a solicited response for the request, or if it is an unsolicited response.

In addition, when a sequence loses its own function of detecting duplicate and loss segments when ignoring a sequence, we need a recovery measure that considers this point.

3.3 Recovery Measure

Our proposed recovery measure can be divided into four steps: 1) routinization of network characteristics to determine and routinize the characteristics of the DNP3 network to be recovered, 2) identification of the first segment including payload header, 3) identification of mid-segments to identify and sort the mid-payload segments, and 4) final segment identification and reassembly to identify the final payload segment and reassemble it, or to identify an abnormal segment and treat it as abnormal.

3.3.1 Routinization of Network Characteristics

The first step prepares the main information required for the process of reassembly. This information includes 1) the application layer header of the data payload for identifying the first segment, 2) the difference between data to distin-

guish the mid-segment, and 3) the number of data points to identify the final segment and to verify the reassembly. The essential information in the application layer header is the Request/Response Object group type. For precise identification, it sometimes needs to store additional information such as a variation and qualifier field. Through the type of object group, it is possible to anticipate the response payload header for a request and to identify the first segment of a response suitable for a request, even when receiving a number of response messages including unsolicited responses.

The information for identifying the mid-segment is required only when a segment is divided into four parts or more. Since the mid-segment includes only the data payload, it is difficult to determine this payload by only examining its structure. Considering that the data of an industrial control system has its own regularity, it is possible to map it partially to the right sequence of a mid-segment that is similar to the previous response segment of data, and it needs to extract and routinize the information on it. This paper does not describe this kind of information extraction, which will be conducted in future research.

The information needed to identify the final segment is a data point address range or the number of data points. When the data point address range is used variably in a request, the information on the data point address range is required. Otherwise, the number of data points is required. With the information on the data point address range or the number of data points, it is possible to determine the number of data points included in the final payload segment. Using the length difference from the previous packet, it is possible to identify the final payload segment, and to compare the total number of received data points, to verify reassembly.

The best measure for this rule is to use the main information by routinizing it after analyzing the data sent/received between the master and an outstation. This is accomplished by using the information on the DNP3 communication operation. However, if it is impossible to routinize by using information on the DNP3 communication operation and samples of communication data to grasp its rules are not enough, it is impossible to recover segments. The reason is the difficulty in determining the class data request answering in a specific situation, or data requests for all usable point lists. Therefore, to distinguish between the mid-segments, it is required to routinize the type of request/response objects, the data range or its number, and the information. This is accomplished by using the information on the DNP3 operation or enough samples of communication data sets.

Table 1 lists key information for the routinization of network characteristics.

3.3.2 Identification of First Segment

In the second step, first it needs to write the information on the main header field of the expected response message by referring to the written rules for the data object group of the request packet. When the received segment is compared with the head information already written, the matched segment is the first segment. Thus, it is possible to distinguish a solicited response from the received unsolicited response because the header information is different. In addition, since first segment's structure is different from that of the data payload segment, it is possible to detect the loss of the first segment when the data payload segment arrives first. By making the master recognize that the sequence information is abnormal when a packet loss is detected, subsequent segments will be discarded. Figure 6 shows a state diagram to identify the first segment.

Table 1 Information for routinization of network characteristics

Category	Field	Location at message
Information for first segment	Function code	13th byte
	Object group type	16th byte
	Variation	17th byte
	Qualifier	18–19th byte
Information for mid-segment	Data difference	Data dependent
Information for final segment	Data point range	19th–20th byte or 19th–22th byte - Variation dependent

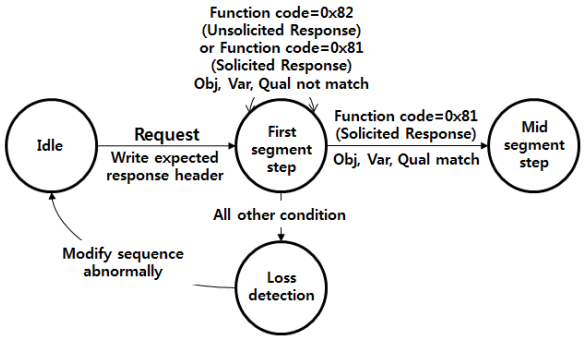

Fig. 6 State diagram to identify the first segment

3.3.3 Step to Identify the Mid-Segment

The third step handles only the data payload without the information in the application layer header. Even when an unsolicited response is received, the segment is unconditionally the segment of a solicited response. The reason is that one device processes many responses in order, rather than processing them simultaneously. When the first segment is identified, the previous unsolicited response has already been processed. Therefore, it is possible to know that the segment is that of the relevant request.

Figure 7 shows a state diagram to identify the mid-segment. First, in order to detect a duplicate segment of the first segment, the segment is processed to confirm whether it is a duplicate segment of the first segment. This is done by referring to the header information in the same way as that of the second step. When a duplicate segment is detected, a reassembly recovery module discards it or makes the master recognize it and wait for the subsequent normal segment.

The length of the field needs to be confirmed. The mid-segment should be confirmed because it is processed with

the treatable maximum length. If its length is not the maximum length, this results in the loss of all mid-segments and the receiving of the final segment. Thus, it makes the master discard the already-received segments and subsequent segments by treating sequences as abnormal.

By using the information on the data differences between the mid-segments, the confirmation of a normal mid-segment is followed by confirmation of whether the segment is in due order. For this, the differences in data values of a specific order can also be used. If a received data point number equals a request data point number minus a maximum point of 1 segment, we move to the next step.

3.3.4 Step to Identify the Final Segment, and Reassembly

Figure 8 shows a state diagram to identify the final segment and reassemble. In the final step, it is possible to calculate the length of a segment by anticipating the number of data points that will be included in the final segment. This is accomplished using the number of response data points for the request corresponding to the rules extracted in the first step. The final segment is identified by the following: when the length of a segment is the maximum length, it is a duplicate segment of the previous segment, and the segment loss means the segment did not arrive within the time limit.

When we conducted an attack to make reassembly impossible by manipulating FIR, FIN, and Sequence fields to be abnormal, the master dropped the packet and changed to idle status, as shown in Fig. 3. This made normal data collection impossible. Therefore, in order to measure the reassembly recovery, the master should modify abnormal fields to normal ones, and transmit them before receiving the packet. Alternatively, the master can force normal communication by modifying the function in the library even when receiving abnormal packets. However, when duplicate segments or segment loss is detected, the existing abnormal processing should be performed.

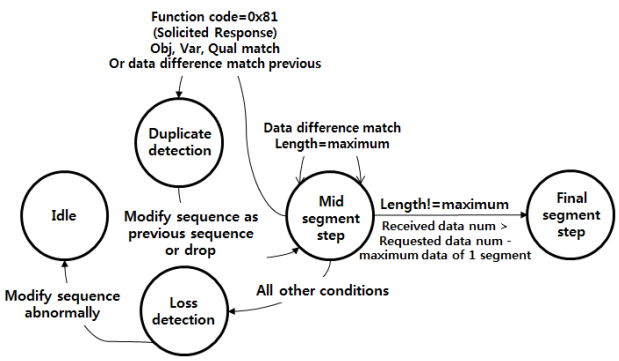

Fig. 7 State diagram for step to identify the mid-segment

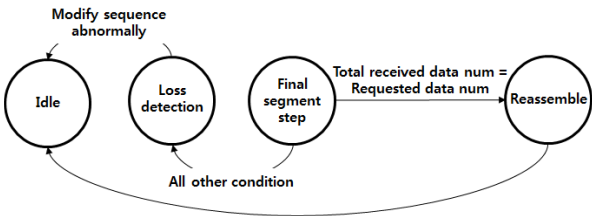

Fig. 8 State diagram to identify the final segment and to reassemble

4. Attack and Reassembly Verification

We conducted two kinds of attacks on the DNP3 reassembly. Both attacks included one unsolicited response before a solicited response. The first attack tested the recovery function by manipulating the Transport Function in general cases. We made normal reassembly impossible by inversing both FIR and FIN.

The second attack tested the original Transport Function's purpose by including missing and duplicate segments. We made segment loss occur by duplicating the second segment and dropping the third segment, in addition to the first attack. Figure 9 shows these attack cases. These two kinds of attacks were conducted against binary input responses composed of four segments for 750 binary input object requests.

The number of data objects and points for reassembly recovery were determined based on actual data collected for two days. Since a fixed range of report data was used, we used not the point range but the point number, and generated four communication rules from three fields: request object type, response object type, and point number. Additionally, for binary input, we stored the statistics for the first point value of the second and third segments to distinguish between the concurrence of a duplicate segment and missing segment. To do this, we used four segments. Table 2 lists the binary input rules and values of the fields of single response attack segments.

4.1 Attack on Transport Function

The recovery for the first attack is as follows. First, as shown in the state diagram of Fig. 10, we should write an expected response header from a request message. The DNP3 request is shown in Fig. 11. Since the request object was a binary input with status, the rule in Table 2 can be used.

The next step is to identify the first segment by inspecting the header of the payload. The 1st DNP3 response message is shown in Fig. 12. We should check the main information, including the function code, which was the 13th byte

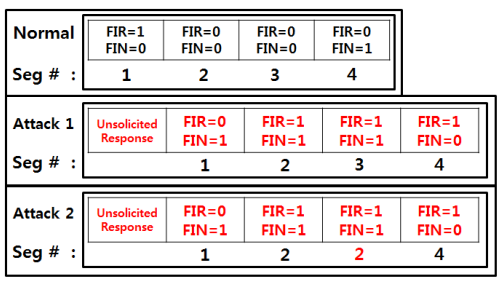

Fig. 9 Multiresponse attack cases

Table 2 Binary input rules and values of multiresponse attack segments

Category		Func Code	Obj type	Var	Qual	Number of items	Difference
Rule		0x81 Response	0x01 B I	0x02	0x0001	0x00–0xED Total 750	Use 1st byte diff
Attack 1	1st msg	0x82	0x02	0x01	0x2801	1	0xCA
	2nd msg	0x81	0x01	0x02	0x0001	238	0xCA
	3rd msg	0x01	0x01	0x01	0x0101	249	0x00
	4th msg	0x01	0x81	0x81	0x8101	249	0x01
	5th msg	0x01	0x81	0x81	0x8181	14	0x01
Attack 2	1st msg	0x82	0x02	0x01	0x2801	1	0xCA
	2nd msg	0x81	0x01	0x02	0x0001	238	0xCA
	3rd msg	0x01	0x01	0x01	0x0101	249	0x00
	4th msg	0x01	0x01	0x01	0x0101	249	0x00
	5th msg	0x01	0x81	0x81	0x8181	14	0x01

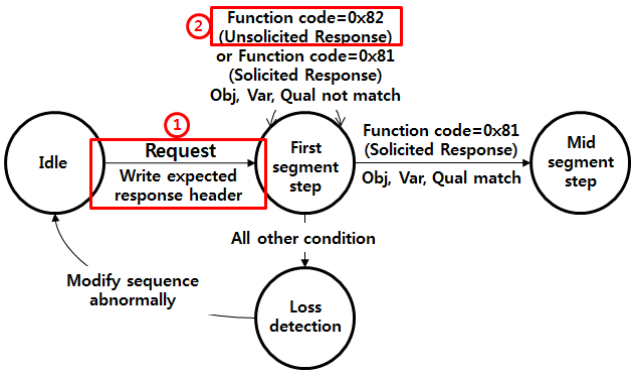

Fig. 10 State diagram for recovery process: 1st message

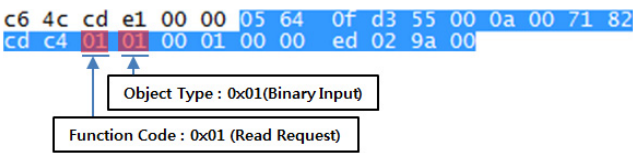

Fig. 11 DNP3 request message: read request

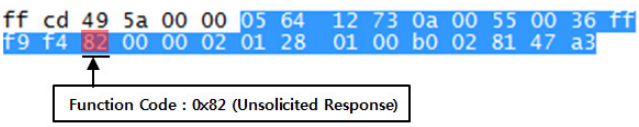

Fig. 12 DNP3 response message: 1st message

of the response message, and the object header, that is, the 16–19th bytes. In the first message, because we could identify that the 13th byte was 0x82, this means the message is an unsolicited message rather than a solicited message. Thus, we skipped this message and moved to the "first segment step" state again.

The 2nd DNP3 response message is shown in Fig. 13. In the second message, the 13th byte was 0x81, which indicates a solicited response. The 16th and 17th bytes were a response message to the binary input status by confirming it was 0x0102. and for more information 18–19th bytes, qualifier code is 0x0001 by big endian, we could identify first segment and recover Transport Function field, this state is

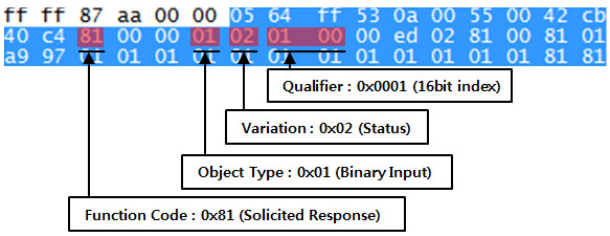

Fig. 13 DNP3 response message: 2nd message (partial)

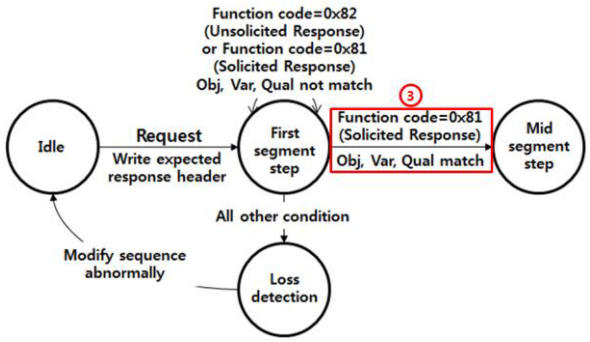

Fig. 14 State diagram for recovery process: 2nd message

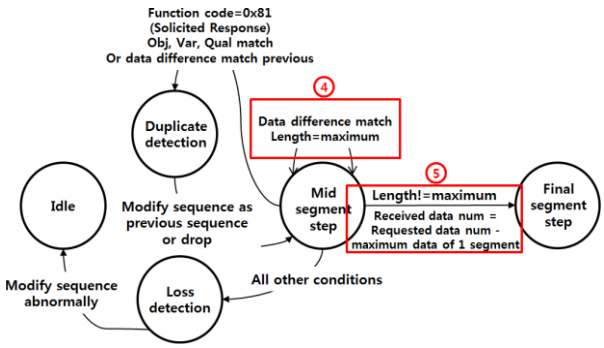

Fig. 15 State diagram for recovery process: 3rd and 4th message

shown in Fig. 14. However, a qualifier code was considered because many values actually reported as binary input status are 0x81 and 0x01. This means there are possibilities for other segments also equal to expecting a response header by accident. Therefore, a qualifier code field was used additionally for certainty.

As shown in the state diagram in Fig. 15, the step involving the identification of the mid-segment needs to include a confirmation as to whether it duplicates one of the first segments. The 3rd DNP3 response message is shown in Fig. 16. The function code of the third message was confirmed as 0x01 (read), which was different from the previous message. Thus, we could identify that it was the segment containing only the data payload, not a duplicate of the first segment.

In addition, considering the third byte, the length field was 0xFF. We knew that it was the maximum-length segment that contained a total of 249 binary objects. We could determine that it was not the final segment but the second

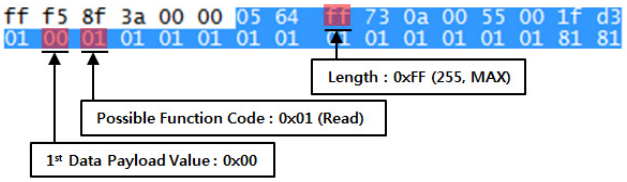

Fig. 16 DNP3 response message: 3rd message (partial)

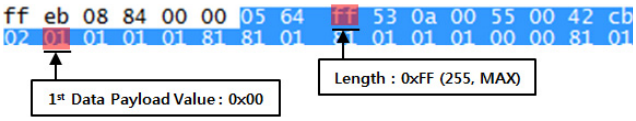

Fig. 17 DNP3 response message: 4th message (partial)

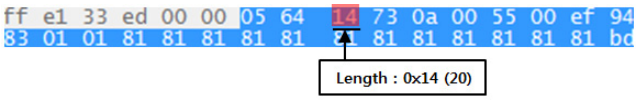

Fig. 18 DNP3 response message: 5th message (partial)

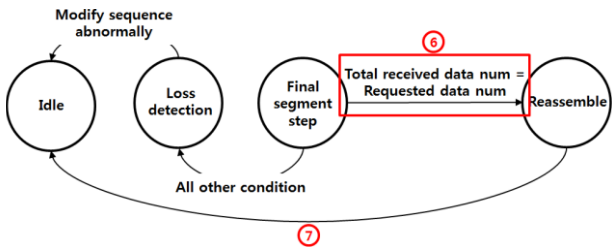

Fig. 19 State diagram for recovery process: 5th message

segment or the third segment after losing the second segment.

The value of the first data point of the second segment (239th data point) was used as 0, and the value of the first data point of the third segment (487th data point) was used as 1. To compare the second segment and the third segment for two days fixedly, we used this information to distinguish the second segment from the third segment. Therefore, we could identify the second segment through the fact that the value of the first data point was 0, and received data is 487 and 487 is less than 511, requested data points(750) minus data points of 1st segment(238), state go to "mid-segment step" state again as Fig. 15's 4th step.

Similarly, we identified the third segment (Fig. 17) through the fact that its length was 255, its first data point was 1, and the received data were 736, which was more than 511. Thus, the state moved to the "final segment" step (the 5th step in Fig. 15).

The 5th DNP3 response message is shown in Fig. 18. For the final segment, as shown in Fig. 19, after identifying that its length was 20 and that it contained 14 data points, the remainder data points (750 requested data points minus 736 data points contained in the previous 3 segments), we could recover the response segment for a total of 750 binary

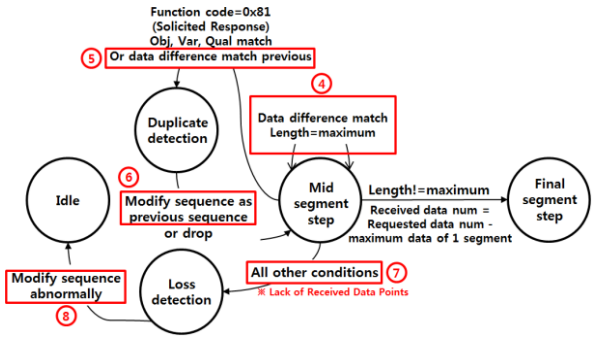

Fig. 20 State diagram for recovery process: 5th message

data points.

4.2 Attack on Transport Function with Message Loss and Duplication

For the second attack, identifying the first and second segments was accomplished in the same manner. However, in the 4th message, we knew that the third segment was a duplicate of the second segment by confirming that the first data point of the third segment was not 1 but 0, equal to second segment. Therefore, we moved to the "duplicate detection" state (Fig. 20, 5th step). We let the master know it was a duplicate segment by modifying it in the same way as before, and let the master wait for the next segment without discarding the segment. We then moved to the "mid-segment step" state again (Fig. 20, 6th step).

However, since the length of the next segment was 20, we could obtain a total of only 487 binary data points. Therefore, we could know that the current response was abnormal in the end, and moved to the "loss detection" state (Fig. 20, 7th step). Accordingly, when transmitting the sequence of the final segment after modifying it as abnormal, the master also discarded two response messages received previously. Then, the state moved to "idle" (Fig. 20, 8th step).

5. Conclusion

An attack that makes reassembly impossible can infringe on the availability with only 1 bit. Since the attack can be conducted without requiring additional information on the data payload, it can be a very attractive option for an attacker. In this respect, a reassembly recovery measure needs to be considered as much as data payload detection. In this paper, we proposed a recovery measure against reassembly attack cases, and verified that it was possible to perform reassembly in attack cases. Existing related works [1]–[7] focused on the detection of cyberattacks but could not ensure availability, which is the most important characteristic in a control system. Therefore, after availability is ensured by using the proposed assembly recovery measure, the detection techniques presented in related works can be applied.

Communication is performed normally for a packet

that has abnormal fields of FIN, FIR and Sequence. However, it is possible that when using four or more segments, we cannot detect them when a duplicate segment and missing segment occur at the same time. Therefore, it is important to study learning mechanisms that can distinguish the mid-segment effectively.

Acknowledgments

This work was supported by the Power Generation and Electricity Delivery Core Technology Program of Korea Institute of Energy Technology Evaluation and Planning (KETEP) granted financial resource from the Ministry of Trade, Industry and Energy, Republic of Korea (no. 20131020402090) and MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-2016-0-00304) supervised by the IITP (Institute for Information & communications Technology Promotion).

References

[1] T. Mander, F. Nabhani, L. Wang, and R. Cheung, "Data object based security for DNP3 over TCP/IP for increased utility commercial aspects security," 2007 IEEE Power Engineering Society General Meeting, pp.1–8, 2007.
[2] A. Carcano, I.N. Fovino, and M. Masera, "Modbus/DNP3 state-based filtering system," 2010 IEEE International Symposium on Industrial Electronics (ISIE), pp.231–236, 2010.
[3] H. Lin, A. Slagell, C.D. Martino, Z. Kalbarczyk, and R.K. Iyer, "Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol," Proc. 8th Annual Cyber Security and Information Intelligence Research Workshop, 2013.
[4] J.-H. Yun, et al., "Burst-based anomaly detection on the DNP3 protocol," Int. J. Contr. Autom. 6.2, pp.313–324, 2013.
[5] J. Nivethan and M. Papa, "A Linux-based firewall for the DNP3 protocol," 2016 IEEE Symposium on Technologies for Homeland Security (HST), pp.1–5, 2016.
[6] A. Shahsavari, A. Sadeghi-Mobarakeh, E. Stewart, and H. Mohsenian-Rad, "Dittribution Grid Reliability Analysis Considering Regulation Down Load Resources Via Micro_PMU Data," 2016 IEEE International Conference on Smart Grid Communications (SmartGridComm), pp.472–477, 2016.
[7] H. Xu, H. Huang, R.S. Khalid, and H. Yu, "Distributed Machine Learning based Smart-grid Energy Management with Occupant Cognition," 2016 IEEE International Conference on Smart Grid Communications (SmartGridComm), pp.491–496, 2016.
[8] Electric Power Research Institute, Distributed Network Protocol 3 Security Interoperability Testing, Palo Alto, CA, Dec. 2011,
[9] S. East, J. Butts, M. Papa, and S. Shenoi, "A Taxonomy of Attacks on the DNP3 Protocol," Critical Infrastructure Protection III, vol.311, pp.67–81, Springer Berlin Heidelberg, 2009.
[10] IEEE Standard for Electric Power Systems Communications Distributed Network Protocol (DNP3), IEEE Standard Association, IEEE Std 1815-2010, 2010.07.01.
[11] IEEE Standard for Electric Power Systems Communications Distributed Network Protocol (DNP3), IEEE Standard Association, IEEE Std 1815-2012, 2012.10.10.
[12] M. Majdalawieh, F. Parisi-Presicce, and D. Wijesekera, "DNPSec: Distributed network protocol version 3 (DNP3) security framework," Advances in Computer, Information, and Systems Sciences, and Engineering, Springer, pp.227–234, 2007.
[13] G. Gilchrist, "Secure authentication for DNP3," 2008 IEEE Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century, pp.1–3, 2008.
[14] D. Lee, et al., "Simulated attack on dnp3 protocol in scada system," Proc. 31st Symposium on Cryptography and Information Security, Kagoshima, Japan. 2014.
[15] H. Yoo and T. Shon, "Challenges and Research Directions for Heterogeneous Cyber Physical System based on IEC 61850: Vulnerability, Security Requirement, and Security Architecture," Future Generation Computing System, vol.61, pp.128–136, Aug. 2016.

Sungmoon Kwon received B.S. degree in Information and Computer Engineering from Ajou University, Suwon, Korea, in 2012. He is currently pursuing his Ph.D. in Computer Engineering from Ajou University, and is with the Information and Communication Security Lab. His current research interests are Smart Grid Security and Industrial Control System Security.

Hyunguk Yoo received B.S. degree in Information and Computer Engineering from Ajou University, Suwon, Korea, in 2012. He is currently pursuing his Ph.D. in Computer Engineering from Ajou University, and is with the Information and Communication Security Lab. His current research interests are Smart Grid Security and Industrial Control System Security.

Taeshik Shon received his Ph.D. degree in Information Security from Korea University, Seoul, Korea and his M.S. and B.S. degree in computer engineering from Ajou University, Suwon, Korea. While he was working toward his Ph.D. degree, he was awarded a KOSEF scholarship to be a research scholar in the Digital Technology Center, University of Minnesota, Minneapolis, USA, from February 2004 to February 2005. From Aug. 2005 to Feb. 2011, Dr. Shon had been a senior engineer in the Convergence S/W Lab, DMC R&D Center of Samsung Electronics Co., Ltd. He is currently a professor at the Division of Information and Computer Engineering, College of Information Technology, Ajou University, Suwon, Korea. He was awarded the Gold Prize for the Sixth Information Security Best Paper Award from the Korea Information Security Agency in 2003, the Honorable Prize for the 24th Student Best Paper Award from Microsoft-KISS, 2005, the Bronze Prize for the Samsung Best Paper Award, 2006, and the Second Level of TRIZ Specialist certification in compliance with the International TRIZ Association requirements, 2008. He is also serving as a guest editor, an editorial staff and review committee of Computers and Electrical Engineering - Elsevier, Mobile Network & Applications - Springer, Security and Communication Networks - Wiley InterScience, Wireless Personal Communications - Springer, Journal of The Korea Institute of Information Security and Cryptology, IAENG International Journal of Computer Science, and other journals. His research interests include Industrial Control System, Anomaly Detection Algorithms, and Digital Forensics.