Design of fuzzing test tool based on WIA-PA protocol

Haofei Xie, Zhihui Wang* and DengXin Yang

Automated institute, Chongqing University of Posts and Telecommunications, Chongqing, 400065, China

*Corresponding author’s e-mail: s180331040@stu.cqupt.edu.cn

Abstract. To solve the problem that the existing fuzzing test tools can not directly test the WIA-PA protocol stack, this paper adds the fuzzing test function of WIA-PA protocol on the basis of the WIA-PA protocol conformance test platform, and integrated it into the conformance test platform in the form of a tool. The essence of fuzzing test for WIA-PA protocol is to test specific functions represented by different command identifiers. The test tool designed in this paper is used to carry out fuzzing test on the WIA-PA protocol stack which has passed the protocol conformance test. The test results show that the test tool designed in this paper has a certain ability of vulnerability detection.

1. Introduction
With the rapidly development of network technology, the networking and informatization of industrial systems have greatly promoted industrial production efficiency, and at the same time, they have brought many potential security risks. Fuzzing test is a testing technique that influences internal program execution through external input data. Its basic principle is to inject the error data that may cause the problem of the program into the application program, observe the operation result, and mine the weak point of the software through the error of the program operation [1]. Existing industrial wireless protocol fuzzing test mainly has the following two problems: a) Existing fuzzing test tools such as Peach, Sully and SPKIE can not directly test industrial wireless protocol. b) Lack of a unified description model for industrial wireless protocols, resulting in randomly generated fuzzing data that does not meet the semantic constraints between fields and the generated fuzzing data has high redundancy [2]. Therefore, this paper first based on the WIA-PA protocol which is one of the three major industrial wireless protocols, and expanded the WIA-PA protocol fuzzing test function on the basis of the WIA-PA protocol conformance test platform, and integrated it into the conformance test platform in the form of a tool. Then a five-tuple model is introduced to describe the WIA-PA protocol data frame [3], and the five-tuple model is used to construct protocol scripts which is used for fuzzing test tool.

2. Design of fuzzing test tool

2.1. Overall architecture
The test tool is integrated into the platform by implementing the WIA-PA protocol conformance test platform extension interface. When the actual test task is executed, the fuzzing test tool and the benchmark test device jointly complete the test task. The benchmark test equipment is the existing test device of WIA-PA protocol conformance test platform, and integrated it into the conformance test platform in the form of a tool. Then a five-tuple model is introduced to describe the WIA-PA protocol data frame [3], and the five-tuple model is used to construct protocol scripts which is used for fuzzing test tool.
The fuzzing test tool component includes six parts: protocol script analysis component, data fuzzy component, fuzzy data generation component, fuzzy data execution component, heartbeat detection component, and test log recording component. The use of the fuzzing test tool starts from the human-computer interaction interface, including five visual interfaces for new protocol model, test execution management, fuzzing strategy selection, testing process monitor and test log query.

2.2. Fuzzing test tool components working relationship

The new protocol model is the entry point for the entire test tool to start work. The constructed protocol script is imported into the test tool through the visual interface. The construction method of the protocol script is implemented according to the principle of the five-tuple model introduced in introduction, so the data frames of WIA-PA protocol are divided into fine granularity. The protocol script analysis component dynamically generates a visual protocol tree model interface based on the script content. The data fuzzy component and the fuzzy data generation component call the mutation factor that meets the constraint conditions according to the parameters passed by the visualization interface and generate complete test data. The fuzzy data execution component completes the sending and receiving of testing data by implementing the communication interface. The heartbeat detection component determines whether the device under test is abnormal by sending normal request data. The test log recording component is responsible for recording abnormal test data. The working relationship of the fuzzing test tool component is shown in Figure 2.

Figure 1. Overall technical architecture

Figure 2. Fuzzing test tool components working relationship
2.3. Design of key component

2.3.1. Design of data fuzzy component
Data fuzzy component queries the node information in the node object dictionary according to the mutable node Id. Node information includes node length, initial data, data type, and valid value range. The mutator calls available mutation functions from mutation library function which according to the constraints provided by node information, and stores multiple fuzzy data into mutation DataSet. Mutation library function were established by CVDN[4] and WIA-PA protocol, which included 8 mutation functions. The data combiner outputs the mutation coverage set through the Cartesian product. The fuzzy data generation component replaces the leaf nodes of the protocol tree according to the mutation coverage set, and obtains complete test data through post-order traversal. The data fuzzy component design is shown in Figure 3.

2.3.2. Design of fuzzy data execution component
If the data transmission rate of the test tool is greater than the processing rate of the device under test, the test results may be inaccurate. In order to minimize the influence of the test tool itself on the test results, it is necessary to design the data cache during the test execution. The fuzzy test data queue and cache area are mainly used to store and process multiple fuzzy test data issued by the test execution component during the test process, as well as the corresponding test response data uploaded by the tested device. The performance monitor is responsible for monitoring the number of messages in the cache, and managing the cache space. The thread manager ensures that data can be sent or received error-free because of test tool communicates with the benchmark device via a serial port. Design of fuzzy data execution component is shown in Figure 4.

2.3.3. Design of heartbeat detection component
Since the WIA-PA protocol stack is running in the radio frequency module, it is not possible to use a third-party monitor to observe the operation of the tested protocol stack. The operation of the tested protocol stack can only be judged by the test response. Due to the characteristics of fuzzy test, most generated fuzzy test data cannot stimulate the tested protocol stack, so it is necessary to design a heartbeat detection component to judge the running condition of the tested protocol stack by sending normal request data. Design of heartbeat detection component is shown in Figure 5.
Figure 5. Design of heartbeat detection component

The heartbeat response timeout is detected after sending fuzzing test data T5. The fuzzing test data T5 that causes a abnormal is recorded and then send T5 separately. If no abnormality is detected, the traversal depth will be increased by one and T1, T5 will be sent. If no abnormality is detected, the traversal depth will continue to be increased and T1, T2, T5 will be sent. At this time, the heartbeat response timeout is detected, so the fuzzing test data causing abnormality will be listed as T1, T2 and T5.

3. Test verification

The target of this test verification is the WIA-PA protocol stack device that has passed the protocol conformance test. This test carried out on the above device according to the fuzzing test process, and analyzed the vulnerability existing in the implementation of WIA-PA protocol stack.

3.1. Determine the test entrance

According to the topology of the test environment shown in Figure 6, the device under test can be one of the gateway device, routing device, or field device. WIA-PA network layer provides 32 different command identifiers such as join request, join response, link increase request, link increase response and so on. Due to space limitation, this paper selected representative unstructured attribute acquisition command identifier of field device as the test entrance.

Figure 6. Topology of the test environment
3.2. Perform fuzzing test

Use PacketSniffer to capture the interactive message when the field device and the gateway device communicate normally, and combine the WIA-PA protocol specification analysis to obtain the unstructured attribute acquisition data message format of the field device. The protocol data script is constructed according to the data message format and imported into the test tool to generate the protocol tree modeling interface. Then select sequential mutation strategy to generate test data and click BEGIN button to execute test. The testing process monitor is shown in Figure 7.

3.3. Test result analysis

After fuzzing test, 58 abnormalities were found through test log. Table 1 show the test result statistics of abnormal fields. Abnormal judgment conditions: a) Abnormal heartbeat detection. b) After sending the fuzz test data, the command execution result bit is not set to be one. According to the WIA-PA protocol specification, the flag is set to be zero if the command execution result is successful, and flag is set to be one if it fails.

| Field Type               | Number of test data | Number of abnormal |
|-------------------------|---------------------|--------------------|
| Frame control field     | 275                 | 7                  |
| Payload length field    | 51                  | 51                 |

Through further analysed of frame control abnormal, when frame control field were 0X03, 0X1F, 0X21, 0X23, 0X25, 0X27, 0X3F, the field device actively leaved the current network and was not monitoring the Beacon frame sent by the test tool. Analysed the above seven abnormal data, when b0 bit of the frame control is one, it causes the device under test to be abnormal. Combined with the WIA-PA protocol specification, position one of b0 is the identifier of the aggregated packet. Therefore, when the frame control field was an aggregated packet, and the payload was a non-aggregated packet, the field device would be down. From the above analysed, it can be included that the frame control field failed to correctly handled the abnormal caused by the fuzzing test data and the field device would be down. So there is a denial of service vulnerability in this field.

Through further analysed of the abnormal payload length, no matter whatever the value of the payload length was, as long as the payload content was correct, the device under test could respond the test data correctly. Therefore, it can be concluded that the equipment under test did not judge the validity of the payload length.
4. Conclusion
Based on the analysis of the existing problems in the industrial wireless protocol fuzzing test, this paper designed and implemented the fuzzing test tool on the basis of the WIA-PA protocol conformance test platform, and made detail design of the overall architecture and components of the "WIA-PA protocol fuzzing test tool". Using this tool to performed fuzzing test on the device under test, it was found that the device under test had a denial of service vulnerability and an implementation abnormal in the protocol stack, which proved the effectiveness of the fuzzing test tool. In the future, we can further combine the characteristics of the WIA-PA protocol to design mutation factors to increase the probability of vulnerability mining.

Acknowledgments
This work was financially supported by 2019 Industrial Technology Foundation Public Service Platform Project "Public Service Platform Construction for Standard Verification and Testing in the Field of Internet of Things" (No.2019-00894-1-1).

References
[1] J. Gao et al., (2020) EM-Fuzz: Augmented Firmware Fuzzing via Memory Checking, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 11, pp. 3420-3432.
[2] H. Liang, X. Pei, X. Jia, W. Shen and J. Zhang., (2018) Fuzzing: State of the Art. IEEE Transactions on Reliability, vol. 67, no. 3, pp. 1199-1218.
[3] Rui Ma, Wendong Ji, Changzhen Hu, Chun Shan and Wu Peng. (2014) Fuzz testing data generation for network protocol using classification tree. In: Communications Security Conference. Beijing. pp. 1-5.
[4] H. Li, W. Zhang, W. Zhou and B. Su. (2014) A Novel Vulnerability Detection Method for ZigBee MAC Layer. In: International Conference on Dependable, Autonomic and Secure Computing. Dalian. pp. 121-124.
[5] J. Li et al., (2018) An integration testing framework and evaluation metric for vulnerability mining methods, China Communications, vol. 15, no. 2, pp. 190-208.