International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 8, Number 3, Year 2026
DOI: https://doi.org/10.62051/ijcsit.v8n3.14
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Study on Artificial Intelligence-Based Network Protocol Fuzz Testing Technology

Xianhang Shang

Fengtai Headquarters Base, Beijing, China

ABSTRACT

With the rapid development of the Internet of Things (IoT), cloud computing, and 5G communication technologies, network protocols, as the core of network communication, are facing increasingly severe security threats. Fuzz testing is an effective technology for detecting vulnerabilities in network protocols; however, traditional fuzz testing methods have limitations such as low test efficiency, blind test case generation, and difficulty in adapting to complex and diverse network protocols. In recent years, the rapid advancement of artificial intelligence (AI) technology has brought new opportunities for the innovation and development of network protocol fuzz testing. This paper systematically studies the application of AI technology in network protocol fuzz testing. First, it elaborates on the urgency of the current network security situation and the importance of network protocol fuzz testing. Then, it introduces the basic principles, core processes, and development stages of network protocol fuzz testing technology. Next, it focuses on the application methods of AI technologies such as machine learning, deep learning, and reinforcement learning in network protocol fuzz testing, and analyzes the improvement effects of AI on fuzz testing efficiency, coverage rate, and vulnerability detection accuracy. Subsequently, it explores the practical application scenarios of AI-based network protocol fuzz testing. Finally, it summarizes the current deficiencies of AI-based network protocol fuzz testing technology and looks forward to its future development trends. This study provides a theoretical reference and practical guidance for the research and application of network protocol fuzz testing technology in the AI era, helping to improve the security and reliability of network protocols.

KEYWORDS

Artificial Intelligence; Network Protocol; Fuzz Testing; Vulnerability Detection; Machine Learning

1. INTRODUCTION

In the digital age, network technology has penetrated into all aspects of social production and life, and network protocols have become the cornerstone for the stable and secure operation of network systems. From the traditional TCP/IP protocol suite to emerging IoT communication protocols (e.g., MQTT, CoAP) and industrial control protocols (e.g., Modbus, DNP3), network protocols undertake the important task of data transmission and interaction between different devices and systems. However, with the continuous expansion of network scale and the increasing complexity of protocol types, the security risks of network protocols have become increasingly prominent. Malicious attackers often exploit vulnerabilities in network protocols to launch attacks such as data theft, service interruption, and system intrusion, which pose a serious threat to national information security, enterprise data assets, and personal privacy.

Fuzz testing, also known as fuzzing, is a widely used vulnerability detection technology. It detects potential vulnerabilities in the tested system by generating a large number of non-standard, unexpected test cases, inputting them into the target system, and then monitoring the system's running

status (e.g., crashes, exceptions, and memory leaks). Compared with other vulnerability detection technologies (e.g., static analysis and dynamic analysis), fuzz testing boasts advantages such as simple principles, high automation, and strong practicality, making it the core technology for network protocol vulnerability detection. However, traditional fuzz testing technologies still have obvious limitations in practical applications: first, test case generation is mostly based on random mutation or simple rule-based generation, which is highly blind and inefficient, resulting in a large number of redundant test cases and low vulnerability detection efficiency; second, traditional fuzz testing struggles to adapt to the dynamic changes and complexity of network protocols, especially for encrypted and proprietary protocols, making it difficult to generate effective test cases; third, the analysis of test results relies heavily on manual judgment, which is not only time-consuming and labor-intensive but also prone to missing potential vulnerabilities [1].

In recent years, with the rapid development of AI technology, machine learning, deep learning, and reinforcement learning have been widely applied in the field of network security, bringing new technological breakthroughs for network protocol fuzz testing. Through data training, AI technology can learn the characteristics and rules of network protocols, enabling intelligent test case generation, adaptive test strategy adjustment, and automatic test result analysis—effectively addressing the shortcomings of traditional fuzz testing technologies. For example, AI-based fuzz testing tools can automatically infer the format of network protocols, generate test cases that are more consistent with protocol specifications, and improve the coverage rate of vulnerability detection. At the same time, AI can real-time analyze the running status of the tested system, quickly identify abnormal behaviors, and improve the efficiency of vulnerability detection. Therefore, studying the application of AI technology in network protocol fuzz testing has important theoretical value and practical significance for improving the level of network protocol security and responding to increasingly complex network security threats.

This paper is structured as follows: Chapter 1 introduces the research background, significance, and current research status; Chapter 2 elaborates on the basic principles, core processes, and development stages of network protocol fuzz testing technology; Chapter 3 focuses on the application methods of AI technology in network protocol fuzz testing and analyzes its application effects; Chapter 4 explores the practical application scenarios of AI-based network protocol fuzz testing; Chapter 5 summarizes the current deficiencies of the technology and looks forward to its future development trends; finally, the full text is summarized.

2. FUZZ TESTING TECHNOLOGY AND ITS DEVELOPMENT

Network protocol fuzz testing is a vulnerability detection technology specifically designed for network protocols. [2] It takes network protocol messages as the test object, generates abnormal or non-standard protocol messages through a series of methods, sends them to the tested protocol entity, and judges whether there are vulnerabilities in the protocol by monitoring the response of the tested entity. The core goal of network protocol fuzz testing is to identify potential vulnerabilities in the protocol design or implementation process, such as buffer overflow, format string vulnerabilities, and logic errors, thereby providing a basis for protocol security reinforcement.

2.1. Basic Principles and Core Processes of Network Protocol Fuzz Testing

The basic principle of network protocol fuzz testing is based on the "fault injection" concept: by simulating an attacker's abnormal input, it causes the tested protocol entity to produce abnormal behaviors, and then identifies vulnerabilities through the analysis of these abnormal behaviors. Unlike traditional software fuzz testing, network protocol fuzz testing must fully consider the characteristics of network protocols, such as protocol format, state transition, and interaction process, making the test process more complex.

The core process of network protocol fuzz testing mainly includes four stages: test preparation, test case generation, test execution, and result analysis. In the test preparation stage, it is necessary to clarify the test object (e.g., a specific network protocol or protocol stack), collect relevant protocol information (e.g., protocol specifications, message format, and interaction process), and build a test environment (including the tested device, test tool, and monitoring tool). In the test case generation stage, a large number of test cases are generated through random mutation, rule-based generation, or other methods based on the collected protocol information; these test cases should cover various abnormal scenarios of the protocol, such as abnormal message length, illegal field values, and incorrect state transitions. In the test execution stage, the generated test cases are sent to the tested protocol entity in sequence, and the running status of the tested entity (e.g., whether it crashes, whether the response is abnormal, and whether there is memory leakage) is monitored in real time. In the result analysis stage, the monitored test results are sorted out and analyzed, false positives are eliminated, real vulnerabilities are confirmed, and the vulnerability types and causes are recorded to provide a basis for vulnerability repair.

Figure 1 shows the framework of network protocol fuzz testing technology, which systematically integrates the core elements and processes of fuzz testing, clearly reflecting the logical relationship between each link.

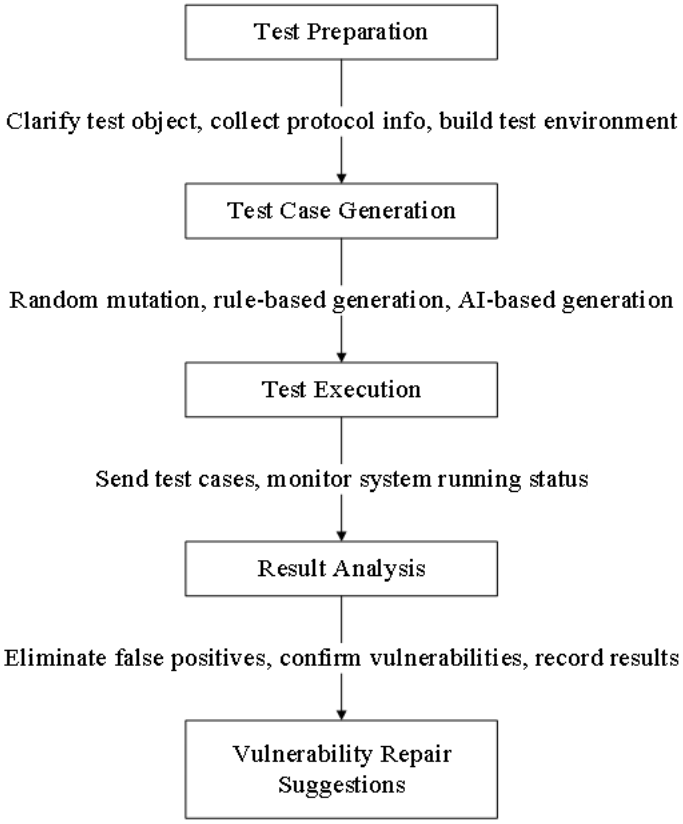

Figure 1. Framework of Network Protocol Fuzz Testing Technology

2.2. Development Stages of Network Protocol Fuzz Testing Technology

With the continuous development of network technology and vulnerability detection needs, network protocol fuzz testing technology has gone through three main development stages, showing a trend of gradual intelligence.

The first stage is the random fuzz testing stage. This stage marks the initial phase of fuzz testing technology. The test case generation method is mainly random mutation, that is, randomly modifying normal protocol messages to generate abnormal test cases. Representative tools of this stage include PROTOS [3] and Peach. The advantages of random fuzz testing are its simple implementation and

low development cost; however, its disadvantages are obvious: test cases are highly blind, redundancy is high, the coverage rate of protocol vulnerabilities is low, and it is difficult to detect deep-seated vulnerabilities. In addition, fuzz testing at this stage does not consider the state transition characteristics of network protocols, making it easy to miss vulnerabilities related to protocol states.

The second stage is the rule-based fuzz testing stage. With the deepening of research on network protocols, researchers began to generate test cases based on rules combined with protocol specifications. This method first parses the protocol format and interaction rules, then generates test cases according to predefined rules (e.g., modifying key protocol fields, violating protocol interaction sequences), which effectively improves the pertinence of test cases and the coverage rate of vulnerabilities. Representative tools of this stage include AFLNet [4]. Compared with random fuzz testing, rule-based fuzz testing has higher test efficiency and vulnerability detection accuracy, but it also has limitations: it relies heavily on the completeness of protocol specifications, and for proprietary or encrypted protocols without clear specifications, it is difficult to formulate effective test rules; in addition, the update of test rules lags behind the iteration of network protocols, making it difficult to adapt to the rapid development of network protocols.

The third stage is the intelligent fuzz testing stage. With the rise of AI technology, researchers began to introduce AI technology into network protocol fuzz testing, realizing the intelligence of test case generation, test strategy adjustment, and result analysis. Fuzz testing at this stage can automatically learn the characteristics and rules of network protocols through data training, without relying on manually formulated test rules, and can adapt to the dynamic changes and complexity of network protocols. Representative tools of this stage include DeepFuzz [5]and ChatAFL [6]. Intelligent fuzz testing has become the mainstream development direction of network protocol fuzz testing technology, effectively solving the deficiencies of traditional fuzz testing and greatly improving the efficiency and accuracy of vulnerability detection.

3. METHODS OF ARTIFICIAL INTELLIGENCE-BASED NETWORK PROTOCOL FUZZ TESTING TECHNOLOGY

The core of AI-based network protocol fuzz testing is to use AI technology to optimize each link of fuzz testing, realize the intelligence of test case generation, test execution, and result analysis, and improve the efficiency and effect of vulnerability detection. Currently, the AI technologies widely used in network protocol fuzz testing mainly include machine learning, deep learning, and reinforcement learning. Different AI technologies have different application methods and effects in fuzz testing.

3.1. Application of Machine Learning in Network Protocol Fuzz Testing

Machine learning is a core technology in the field of AI, which can learn hidden rules and characteristics from a large amount of data and make predictions and decisions based on the learned rules. In network protocol fuzz testing, machine learning is mainly used for test case generation and vulnerability prediction, which can effectively improve the pertinence and efficiency of test cases.

In terms of test case generation, machine learning can first collect a large number of normal network protocol messages, then use clustering algorithms (e.g., K-means, DBSCAN) to cluster the message data and extract the key characteristics of protocol messages (e.g., message length, field type, field value range). On this basis, classification algorithms (e.g., SVM, random forest) are used to generate test cases that are more likely to trigger vulnerabilities.Compared with traditional rule-based test case generation methods, machine learning-based test case generation can adapt to unknown or proprietary protocols without relying on protocol specifications, and the generated test cases have higher pertinence and vulnerability triggering probability.

In terms of vulnerability prediction, machine learning can collect historical vulnerability data of network protocols (e.g., vulnerability types, vulnerability locations, and trigger conditions), build a vulnerability prediction model through training, and predict the potential vulnerability locations and types of the tested protocol. This can help testers focus on the key parts of the protocol, reduce the scope of fuzz testing, and improve test efficiency. For example, researchers have used decision tree algorithms to train the vulnerability data of the TCP/IP protocol and predict the vulnerability locations of the protocol according to the characteristics of protocol messages, which effectively improves the efficiency of vulnerability detection.

3.2. Application of Deep Learning in Network Protocol Fuzz Testing

Deep learning is a branch of machine learning that realizes feature learning and pattern recognition through multi-layer neural networks. Compared with traditional machine learning, deep learning has stronger feature extraction capabilities and can automatically learn high-level abstract features from complex data, making it more suitable for processing complex network protocol data (e.g., encrypted protocol messages, multi-state protocol interaction data).

In network protocol fuzz testing, deep learning is mainly used for protocol format inference, test case generation, and abnormal behavior detection. In terms of protocol format inference, for encrypted or proprietary protocols without clear specifications, deep learning models (e.g., CNN, RNN, Transformer) can be used to analyze a large amount of protocol interaction data, automatically infer protocol formats, field divisions, and state transition rules, which provides a basis for test case generation. For example, researchers have used RNN models to analyze the sequence characteristics of network protocol messages, automatically infer the state transition rules of the protocol, and generate test cases that violate the state transition rules, which effectively improves the coverage rate of vulnerability detection.

In terms of test case generation, deep learning models (e.g., GAN, VAE) can be used to generate realistic network protocol messages. A Generative Adversarial Network (GAN) consists of a generator and a discriminator: the generator generates fake protocol messages, and the discriminator distinguishes between real and fake messages. Through continuous adversarial training, the generator can generate protocol messages that are highly similar to real messages, which are then mutated to generate test cases. This method can generate test cases that are more consistent with the protocol format, reduce the redundancy of test cases, and improve the efficiency of vulnerability detection. For example, researchers have used GAN to generate MQTT protocol messages, and the generated test cases can effectively trigger vulnerabilities such as buffer overflow in the MQTT protocol stack.

In terms of abnormal behavior detection, deep learning models can be used to monitor the running status of the tested protocol entity in real time, learn the normal running mode of the protocol, and identify abnormal behaviors (e.g., crashes, abnormal responses) by comparing the deviation between the actual running mode and the normal mode. This can realize automatic analysis of test results, reduce manual intervention, and improve the efficiency of vulnerability identification. For example, researchers have used CNN models to analyze the running logs of the protocol entity, automatically identify abnormal log information, and quickly locate potential vulnerabilities.

3.3. Application of Reinforcement Learning in Network Protocol Fuzz Testing

Reinforcement learning is a type of machine learning that learns optimal strategies through interaction with the environment. In reinforcement learning, the agent selects actions according to the current state of the environment, obtains rewards or punishments based on the results of the actions, and then adjusts the strategy to maximize the cumulative reward. In network protocol fuzz testing, reinforcement learning can be used to optimize test strategies, realize adaptive adjustment of test cases, and improve the efficiency of vulnerability detection.

In network protocol fuzz testing, the agent can be regarded as a fuzz testing tool, the environment is the tested protocol entity and the test environment, the action is the selection and sending of test cases, and the reward is the feedback from the tested entity (e.g., whether a vulnerability is found, whether code coverage is improved). The agent continuously adjusts the test strategy through interaction with the environment, selects the test cases that are most likely to trigger vulnerabilities, and maximizes the efficiency of vulnerability detection. For example, researchers have used the Q-learning algorithm to build a reinforcement learning model for network protocol fuzz testing, which can adaptively adjust the mutation strategy of test cases according to feedback from the tested entity, and improve the coverage rate of protocol vulnerabilities and the efficiency of vulnerability detection. In addition, reinforcement learning can also be used to handle the state transition problem of network protocols, automatically learn the state transition rules of the protocol, and generate test cases that are more likely to trigger state-related vulnerabilities.

3.4. Application Effects of AI in Network Protocol Fuzz Testing

The application of AI technology in network protocol fuzz testing has significantly improved the efficiency and effect of vulnerability detection, mainly reflected in three aspects: first, improving test efficiency. AI-based test case generation can reduce redundant test cases, improve the pertinence of test cases, and shorten the test cycle.Second, improving vulnerability detection coverage. AI technology can automatically learn the characteristics and rules of network protocols, generate test cases covering more abnormal scenarios, and detect deep-seated vulnerabilities that are difficult to be found by traditional fuzz testing. Third, reducing manual intervention. AI can realize automatic generation of test cases, automatic monitoring of test processes, and automatic analysis of test results, reducing the workload of testers and improving the accuracy of vulnerability identification.Meanwhile, machine learning algorithms have also been widely used in fault prognosis of network-related equipment, providing a reference for vulnerability prevention.

4. APPLICATION OF ARTIFICIAL INTELLIGENCE-BASED NETWORK PROTOCOL FUZZ TESTING TECHNOLOGY

With the continuous maturity of AI technology, AI-based network protocol fuzz testing technology has been widely applied in various fields, such as the Internet of Things (IoT) [7], industrial control, cloud computing, and mobile communication. At the same time, a number of mainstream AI-based network protocol fuzz testing tools have emerged in the industry, each with distinct characteristics and application scenarios. This chapter will introduce the practical application scenarios of AI-based network protocol fuzz testing.

4.1. Internet of Things (IoT) Field

The IoT field involves a large number of devices and diverse network protocols, such as MQTT, CoAP, and ZigBee. Most IoT devices have limited computing resources and simple protocol stacks, making them prone to security vulnerabilities. AI-based network protocol fuzz testing can effectively detect vulnerabilities in IoT protocols, improving the security of IoT devices. For example, researchers have used deep learning models to fuzz test the ZigBee protocol, detecting vulnerabilities such as buffer overflow and denial of service in the protocol, and providing a basis for the security reinforcement of IoT devices.

4.2. Industrial Control System (ICS) Field

Industrial control systems are widely used in fields such as power, petroleum, and chemicals, and their network protocols (e.g., Modbus, DNP3, IEC 61850) are the core of industrial control communication. The security of industrial control protocols is directly related to the stable operation

of industrial production. AI-based network protocol fuzz testing can adapt to the complexity and real-time requirements of industrial control protocols, detect potential vulnerabilities in the protocol, and avoid industrial safety accidents. For example, researchers have used reinforcement learning to fuzz test the Modbus protocol, automatically generating test cases that are more likely to trigger vulnerabilities, and detecting logic errors and denial of service vulnerabilities in the protocol. In recent years, there have also been many studies on fuzz testing of industrial control protocols based on deep features and reinforcement learning, which have further improved the vulnerability detection effect of industrial control protocols.

4.3. Cloud Computing and Mobile Communication Field

In the cloud computing field, network protocols such as TCP/IP, HTTP, and SSH are widely used, and their security directly affects the security of cloud services. AI-based network protocol fuzz testing can detect vulnerabilities in cloud protocol stacks, improving the security of cloud services. In the mobile communication field, 5G protocols (e.g., NR, NGAP) have the characteristics of high speed, low latency, and large connectivity, and their security is crucial to the stable operation of mobile communication networks. AI-based fuzz testing can adapt to the complexity of 5G protocols, detect vulnerabilities in the protocol design and implementation process, and ensure the security of 5G communication. In addition, AI-based fuzz testing is also used in the field of Trusted Execution Environment (TEE); researchers have proposed a fuzz testing method for TEE kernel API interfaces to detect vulnerabilities in the TEE itself.

5. CURRENT DEFICIENCIES AND FUTURE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE-BASED NETWORK PROTOCOL FUZZ TESTING TECHNOLOGY

Although AI-based network protocol fuzz testing technology has made great progress and been widely applied, it still has some deficiencies in practical applications due to the limitations of AI technology itself and the complexity of network protocols. At the same time, with the continuous development of AI technology and network technology, AI-based network protocol fuzz testing technology will show new development trends.

5.1. Current Deficiencies

5.1.1. Dependence on Training Data

AI-based network protocol fuzz testing relies heavily on a large amount of high-quality training data (e.g., normal protocol messages, vulnerability data). However, in practical applications, it is difficult to obtain sufficient training data for some proprietary or new protocols, which leads to poor performance of AI models and makes it difficult to achieve ideal test effects. In addition, the quality of training data also affects the performance of AI models; if the training data contains noise or errors, it will lead to overfitting or underfitting of the model, reducing the accuracy of vulnerability detection. At the same time, AI models themselves have security risks such as training data contamination, which may affect the reliability of fuzz testing results.

5.1.2. Poor Adaptability to Encrypted Protocols

With the continuous improvement of network security awareness, an increasing number of network protocols adopt encryption technology (e.g., TLS/SSL, IPsec) to ensure the security of data transmission. However, AI-based network protocol fuzz testing has poor adaptability to encrypted protocols: on the one hand, it is difficult to parse encrypted protocol messages and extract effective features; on the other hand, the mutation of encrypted messages is easily identified by the encryption mechanism, making it difficult to trigger vulnerabilities. Currently, most AI-based fuzz testing tools

can only handle unencrypted or weakly encrypted protocols, and there is still a lack of effective solutions for strongly encrypted protocols.

5.1.3. High Hardware Resource Requirements

AI models (especially deep learning models) require a large amount of computing resources for training and inference, which places high demands on the hardware environment of fuzz testing. For small and medium-sized enterprises or individual researchers, it is difficult to bear the cost of hardware resources, which limits the popularization and application of AI-based network protocol fuzz testing technology. In addition, the training time of AI models is long, which also affects the efficiency of fuzz testing.

5.1.4. Lack of Effective Evaluation Standards

At present, there is no unified evaluation standard for AI-based network protocol fuzz testing technology, and the evaluation of test effects is mainly based on indicators such as code coverage, vulnerability detection rate, and test efficiency. However, these indicators are affected by factors such as test objects and test environments, making it difficult to comprehensively and objectively evaluate the performance of AI-based fuzz testing tools. This leads to difficulties for testers in selecting and applying tools. Some research institutions have tried to establish evaluation systems, but there is still a lack of industry-wide consensus.

5.2. Future Development Trends

5.2.1. Integration of Multi-AI Technologies

In the future, AI-based network protocol fuzz testing will tend to integrate multiple AI technologies (e.g., machine learning, deep learning, reinforcement learning, and large language models) to give full play to the advantages of various technologies and improve the effect of vulnerability detection. For example, combining deep learning with reinforcement learning can realize intelligent test case generation and adaptive test strategy adjustment; combining large language models with protocol format inference can improve the ability of AI models to parse unknown protocols. At the same time, the "AI vs. AI" offensive and defensive mode will become a new trend, using AI to simulate intelligent attacks and improve the pertinence of fuzz testing.

5.2.2. Optimization of Encrypted Protocol Fuzz Testing Technology

With the widespread application of encrypted protocols, research on AI-based encrypted protocol fuzz testing will become a key direction. In the future, researchers will focus on solving the problem of parsing encrypted protocol messages, such as using homomorphic encryption technology to process encrypted messages, or using AI models to learn the characteristics of encrypted messages without decryption, thereby realizing the fuzz testing of encrypted protocols. In addition, the combination of symbolic execution and AI technology will also provide a new way for encrypted protocol fuzz testing.

5.2.3. Lightweight AI Model Design

To solve the problem of high hardware resource requirements, lightweight AI model design will become an important development trend. Researchers will use technologies such as model compression and quantization to reduce the computing resource requirements and training time of AI models, making AI-based network protocol fuzz testing technology suitable for more application scenarios (e.g., IoT devices with limited resources). At the same time, the integration of edge computing and AI-based fuzz testing will realize on-site testing of edge devices, improving the timeliness of vulnerability detection.

5.2.4. Establishment of Unified Evaluation Standards

In the future, with the continuous development and popularization of AI-based network protocol fuzz testing technology, the industry will gradually establish unified evaluation standards, including evaluation indicators, evaluation methods, and evaluation processes. This will help testers comprehensively and objectively evaluate the performance of fuzz testing tools, promote the healthy development of the technology, and realize the standardized application of the technology in various fields. In addition, the combination of AI and network simulators (e.g., NS-3, IMUNES) will also improve the authenticity and comprehensiveness of fuzz testing evaluations.

5.2.5. Application in Emerging Network Scenarios

With the development of new network technologies such as 6G, the metaverse, and satellite Internet, new network protocols will continue to emerge, and the security requirements for network protocols will be higher. AI-based network protocol fuzz testing technology will be widely applied in these emerging network scenarios, providing security guarantees for the stable operation of new network systems. For example, in 6G networks, AI-based fuzz testing can detect vulnerabilities in new protocols such as space-air-ground integrated communication protocols, ensuring the security of 6G communication. At the same time, AI can also be used to simulate unknown vulnerability attacks, improving the ability to detect zero-day vulnerabilities.

REFERENCES

[1] Zhang, X., Zhang, C., Li, X., Du, Z., Mao, B., Li, Y., ... & Deng, R. (2024). A survey of protocol fuzzing. ACM Computing Surveys, 57(2), 1-36.

[2] Manès, V. J., Han, H., Han, C., Cha, S. K., Egele, M., Schwartz, E. J., & Woo, M. (2019). The art, science, and engineering of fuzzing: A survey. IEEE Transactions on Software Engineering, 47(11), 2312-2331.

[3] Kaksonen, R., Laakso, M., & Takanen, A. (2001, May). Software security assessment through specification mutations and fault injection. In Communications and Multimedia Security Issues of the New Century: IFIP TC6/TC11 Fifth Joint Working Conference on Communications and Multimedia Security (CMS'01) May 21–22, 2001, Darmstadt, Germany (pp. 173-183). Boston, MA: Springer US.

[4] Pham, V. T., Böhme, M., & Roychoudhury, A. (2020, October). Aflnet: A greybox fuzzer for network protocols. In 2020 IEEE 13th international conference on software testing, validation and verification (ICST) (pp. 460-465). IEEE.

[5] Liu, X., Li, X., Prajapati, R., & Wu, D. (2019, July). Deepfuzz: Automatic generation of syntax valid c programs for fuzz testing. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 1044-1051).

[6] Meng, R., Mirchev, M., Böhme, M., & Roychoudhury, A. (2024, February). Large Language Model guided Protocol Fuzzing. In NDSS.

[7] Chen, J., Diao, W., Zhao, Q., Zuo, C., Lin, Z., Wang, X., ... & Zhang, K. (2018, February). IoTFuzzer: Discovering memory corruptions in IoT through app-based fuzzing. In NDSS (pp. 1-15).