Journal of Computer and Communications, 2022, 10, 1-13
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A DoS Attacks Detection Aglorithm Based on Snort-BASE for Robotic Arm Control Systems

Nana Li[1], Yong Wang[1], Pengfei Shen[2], Shuangfei Li[3], Lin Zhou[1]

[1]College of Science, Shanghai University of Electric Power, Shanghai, China
[2]IBM (China) Investment Co. Ltd., Shanghai, China
[3]Shanghai Cloud Sword Information Technology Co. Ltd, Shanghai, China
Email: nana429256@163.com

How to cite this paper: Li, N.N., Wang, Y., Shen, P.F., Li, S.F. and Zhou, L. (2022) A DoS Attacks Detection Aglorithm Based on Snort-BASE for Robotic Arm Control Systems. *Journal of Computer and Communications*, **10**, 1-13.
https://doi.org/10.4236/jcc.2022.104001

Received: March 11, 2022
Accepted: April 4, 2022
Published: April 7, 2022

Abstract

In response to the frequent safety accidents of industrial robots, this paper designs and implements a safety detection system for robot control. It can perform real-time security detection of robot operations on industrial production lines to improve the security and reliability of robot control systems. This paper designs and implements a robot control system based Snort-BASE for real-time online detection of DoS attacks. The system uses a six-degree-of-freedom robotic arm as an example, uses Snort to record the network communication data of the robot arm control system in real time, and filters the network traffic through self-defined rules, and then uses the BASE analysis platform to achieve security analysis of the network traffic. The solution verifies the effectiveness of online real-time detection of attacks and visualisation of attack records by designing simulated robotic arm and real robotic arm attack experiments respectively, thus achieving the security of network communication of the robot remote control system.

Keywords

Robotic Arms, Remote Control Systems, Cyber Attacks, BASE, Snort, Security Protection

1. Introduction

With the development of Internet technology, more and more auxiliary intelligent devices or control systems are penetrating the service, security and entertainment industries, and robots working on industrial production lines are even more expected to interact with remote control terminals in a timely and accurate manner. However, serious safety incidents with industrial robot control systems

are currently frequent, including: in 2015, a sudden malfunctioning restart of an industrial robot at the Ajin USA factory in Gussetta, Alabama, which resulted in the death of a staff member; and in November 2016, a malfunctioning robot system at a high-tech trade show that broke the glass of its own booth, injuring one person. It is because of the large number of robotic arms currently used in industrial production lines to replace human labour that the safety issues of robot control hidden by such incidents will pose an increasing challenge to the development of industrial robots [1].

Existing research on robotic arms is currently focused on path optimisation algorithms and the real-time nature of teleoperation network communication. For example, in 2016, Meng Shaonan *et al.* addressed the robotic arm motion planning problem in complex environments under the ROS platform, using a 3D model to realise its virtual control and derive information on the position and velocity of each joint during the robotic arm motion, providing a more intuitive method for improving the planning algorithm [2]. In 2018, Wu Likai investigated the large communication latency of the space teleoperation system of the robotic arm and the instability of the tracking performance of the operation system, then he proposed a bilateral PID control algorithm for the remote control system based on the adaptive Smith predictor, in order to achieve effective control of the remote control system [3]. However, all the above studies lacked attention to the security of network communication of robots on industrial production lines and the problem of malicious attacks such as tampering or interception of information during remote operation. In industrial control systems, hackers most often use Dos attacks to compromise the target, flooding the control system with useless information and consuming communication resources, thus crippling the target network and preventing it from completing the intended operation [4]. In order to avoid malicious attackers hijacking the work of a robot arm and completely paralysing its control system, resulting in the inability to complete the intended operation, thus posing an unpredictable risk to industrial production [5]. This paper designs and implements an online security detection system for robots.

1) Industrial firewalls are currently the most common method of effective network protection in the field of industrial control security, but they are a passive defence and are usually powerless against attacks from within [6]. The intrusion detection technology used in this paper, as an active information security technology, can effectively make up for the shortcomings of traditional security protection technologies such as firewalls [7].

2) In this paper, a six-degree-of-freedom robotic arm is used instead of a robot control system, and the security risks during the remote operation of the robot are verified through Dos attack experiments.

3) The intrusion detection system designed in this paper uses Snort to record the network communication data of the robot arm control system in real time and filter the network traffic by self-defined rules.

4) Security analysis of network traffic is implemented by the BASE analysis

console.

5) A simulated robotic arm and a real robotic arm control system are designed to test the feasibility and effectiveness of the design solution respectively.

2. Related Research

Rule-based detection is a common misuse detection method in intrusion detection systems, the most representative of which is the Snort intrusion detection system [8]. As an intrusion detection method that converts intrusions into a dynamic knowledge base, it mainly uses the knowledge base to determine the occurrence of events [9]. Snort is able to capture network traffic in real time, analyse the protocols, match the characteristics of captured illegal traffic and suspicious data with the rule base, and record them to a log file with real-time alerts. Snort can also call different decoding functions to parse the protocol types of captured packets, so that different types of abnormal protocols can be identified and alerted according to different detection rules [10]. The Snort consists of four basic parts [11], as follows:

1) Sniffer: listens to packets in the communication network and analyses them.

2) Pre-processor: analyzes the characteristic behavior of the raw packets before sending them to the detection engine.

3) Detection Engine: a key part of Snort, it receives packets from the pre-processor and analyses them using pre-defined rules, if the detected content is consistent with a rule an alarm event will be generated.

4) Alarm/Logging: if a rule set in the detection engine is matched, an alarm will be triggered and the alarm information can be sent to a log file or recorded in SQL via the trap command.

Snort's architecture is organised in a flexible way, as it can be deployed on either the same host platform or on different physical platforms, depending on the actual environment requirements. If only a test and research environment is required, a single server deployment can be chosen, while if a stable and efficient professional intrusion detection platform is required, then a multi-tier distributed deployment can be chosen.

In this paper, we first set up an attack experiment environment and then choose the best network connection for virtual machines to achieve more efficient attack detection experiments. Next, the configuration of the rule-based intrusion detection system for Snort and BASE is carried out, and after the system is working properly, the corresponding network packet listening, data capture and data analysis are then carried out.

3. The Flow of the Testing System Configuration

The rule-based intrusion detection system based on Snort and BASE designed in this paper uses a three-tier architecture of "sensor-database-analysis platform". The combination of WinPcap as the underlying network interface driver that captures and listens for packets in the network [12], and Snort that cap-

tures the datagram and filters the captured network traffic using rules, constitutes the sensor component of a rule-based intrusion detection system. The combination of the two forms the sensor component of the rule-based intrusion detection system, while BASE acts as the analysis platform for operating the query database to perform real-time security analysis and visualisation of the detected data.

The entire experimental environment was deployed by VMware virtual machines. Firstly, a Windows 7 client virtual machine was built using VMware to implement the required network environment. The sensor component, database component and WEB platform were then deployed separately to implement the complete functionality of the Snort and BASE based rule-based intrusion detection system.

1) Sensor components: Install WinPcap and Snort and modify Snort's settings file snort.conf:

a) Set the rule package path;

b) Set the database connection;

c) Set the dynamic pre-processing library directory;

2) Database components: in order to facilitate debugging and management of MySQL services, two databases, snortdb and snortarc, were created. Among them, snortdb is the database for Snort storage, while snortarc is the archive database for BASE.

3) WEB platform: Select Apache + PHP to build the WEB platform and add support for PHP in the Apache settings file httpd.conf.

4) BASE installation and debugging: create the AG database for BASE, use the http://localhost/base/ address to log in and use BASE, and get real-time attack logs through the analysis console.

4. Simulated Robotic Arm Attack Test

The experiments were carried out using Visual Studio 2013, ABB RobotStudio and PC SDK 5.61 together to configure the environment. The main functions achieved were to write the console using Visual Studio 2013, to write the robot arm control program using RobotStudio and to implement the interface communication using PC. SDK, leaving the mouse trajectory in the panel and making the robot arm reproduce the trajectory movement.

4.1. Simulated Robotic Arm Control System Configuration

This experiment started by establishing communication between Visual Studio 2013 and ABB RobotStudio. Firstly, a simple robot system was built in RobotStudio and a simple program path was added. Secondly, the robot system was built and a simple path was set up, at this point the path generated by way of target points in the workstation needed to be synchronised to the virtual demonstrator, as shown in **Figure 1**. Next, Visual Studio was opened to create a project and introduce the program se. Finally, the pc.sdk dll reference

to complete the Visual Studio forms application project build, as shown in Figure 2.

At this time, the Visual Studio control interface and RobotStudio robot arm to establish communication, through the mouse in the control form to draw motion trajectory, can make the RS end of the robot arm repeat the trajectory movement, as shown in Figure 3.

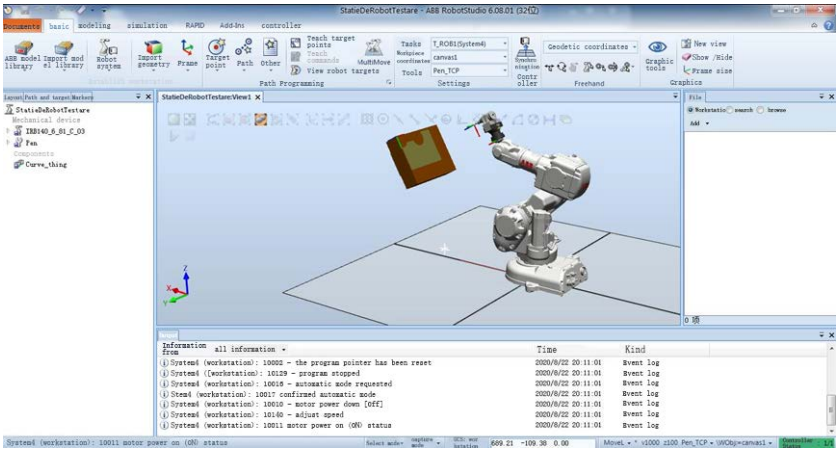


Figure 1. Robotic arm simulation system.

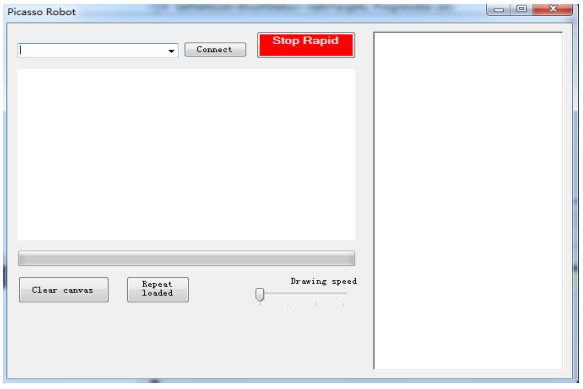


Figure 2. Robotic arm motion control form.

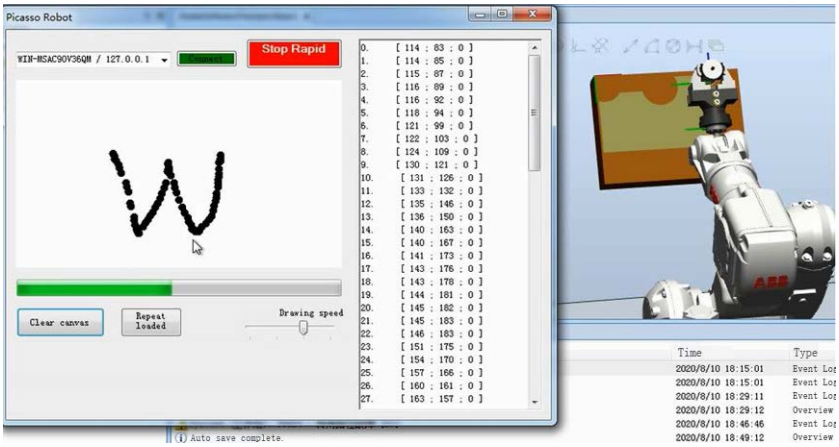


Figure 3. VS establishes virtual communication with RS.

The attack was tested using a Dos attack, which sends a large number of useless TCP packets to the communication process of the simulation environment, with the aim of exhausting the target network or system resources, thus causing the remote control system services of the robotic arm to be temporarily stopped and unable to accept access from normal users.

4.2. Simulated Robotic Arm Attack Experiment

Operating environment: Windows 7 client virtual machine

Snort host IP: 192.168.5.134

Simulated intrusion host IP: 192.168.5.1

Before the attack test, first create alarm rules in the snort file rules directory, the following is an example rule.

alert tcp! 192.168.5.1any → 192.168.5.0/24 21 (content: "USER"; msg: "FTP Login";)

The above rule means: source port number of source ip address → destination port number of destination ip address (rule option), where Alert stands for alarm, tcp stands for protocol type, !192.168.5.1 means the source IP address is not 192.168.5.1, any means the source port is any port 192.168.5.0/24 means the destinat-ion IP range, 21 means the destination port is 21, content: "USER" means the m-atching string is "USER", msg: "FTP Login" means the alarm message is "FTP Login" [13].

The following alarm rules were used for this attack test:

Alert tcp any any → 192.168.5.134/32 any (logto: "log"; msg: "alert tcp !!!!!!"; sid:1000001)

Start snort's intrusion detection mode by entering the command "snort -c c:\ids\snort\etc\snort.conf -l c:\ids\snort\log -d" at the CMD command prompt, the result of which is shown in Figure 4.

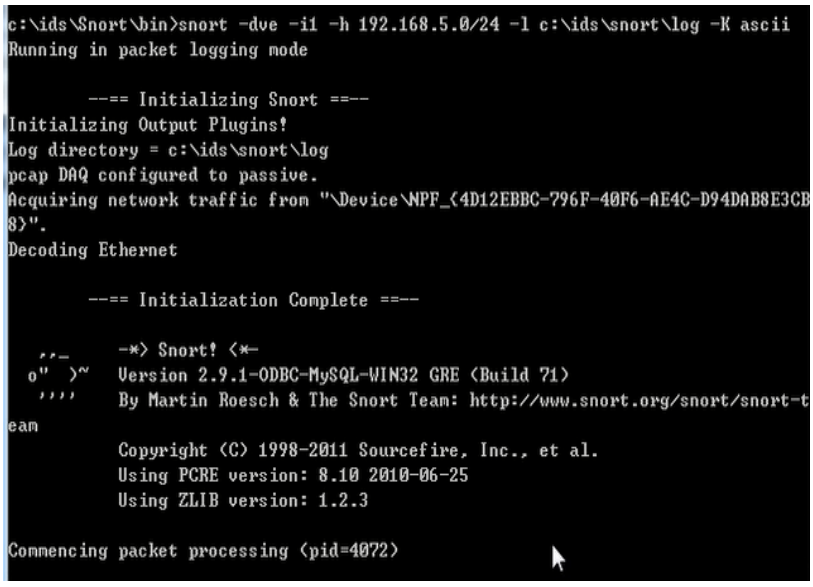


Figure 4. Start snort intrusion detection mode.

With Snort intrusion detection mode on, a Dos attack on the robotic arm simulation environment was written in the python 3.7 development environment, using streaming sockets to establish a connection while ensuring connectivity to the local server and port, and then continuously sending threads to the destination port, exhausting the network or system resources of the target port, causing the service to temporarily stop and fail to accept access from normal users. The result of the attack is shown in Figure 5. The robot arm control window in Visual Studio is down, the mouse is unable to draw motion trajectories, and the robot arm in RobotStudio fails to run and stops moving.

Open the snort file directory, you can see that the log folder has recorded a large number of packet files, as well as containing an alarm text file named CSV, the specific alarm records as shown in Figure 6. Log in to the BASE analysis console to view all the Alert alarm records, which details the information of each attack data, including the type of alarm, attack time, protocol, source IP, destination IP, source port, destination port, etc., as shown in Figure 7.

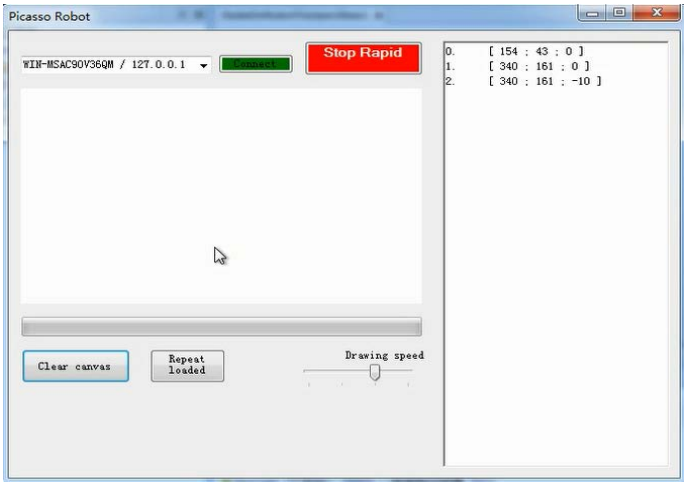


Figure 5. Dos attack simulation robotic arm control system.

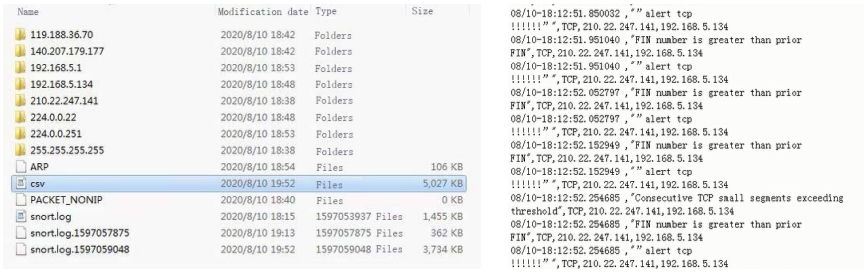


Figure 6. Snort test results.

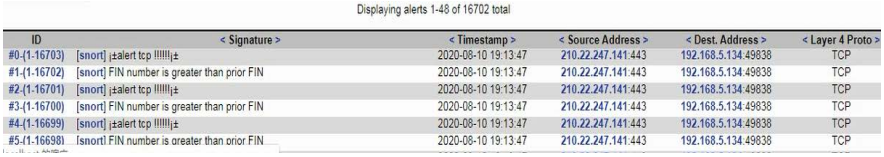


Figure 7. BASE analysis desk attack log.

Simulation results prove that the Snort and BASE based rule-based intrusion detection system can effectively detect attacks and visualize the attack information.

5. Six-Degree-of-Freedom Robotic Arm Physical Attack Test

Based on the results of the attack test described in the previous chapter, this chapter chose to conduct another attack test using a six-degree-of-freedom robotic arm to verify the effectiveness of the Snort and BASE-based rule-based intrusion detection system under a real robot control system, with the robotic arm physically shown in Figure 8.

5.1. Robotic Arm Remote Control System Configuration

1) Servo motor controller

Receives control command signals from the computer to the host computer software and uses them to control the multiple servo servos.

2) Virtual serial port configuration of the robot arm

The virtual serial port is configured via ZLVircom to enable network communication between the robot arm and the computer. The communication process is achieved by a combination of pass-through communication and the virtual serial port.

In the serial port management of the ZLVircom device, select Add Virtual Serial Port COM8 as shown in Figure 9(a). After selecting Add Virtual Serial COM8 in Device Management to configure the connected devices, the information system such as device model, name and ID will be read automatically and the IP address, subnet mask and gateway will be configured to the same network segment as the local IP address. At this point the virtual serial port COM8 has been successfully connected to the device with IP 192.168.1.200, as shown in Figure 9(b), at this point the virtual serial port COM8 has completed the communication configuration with the robotic arm device.

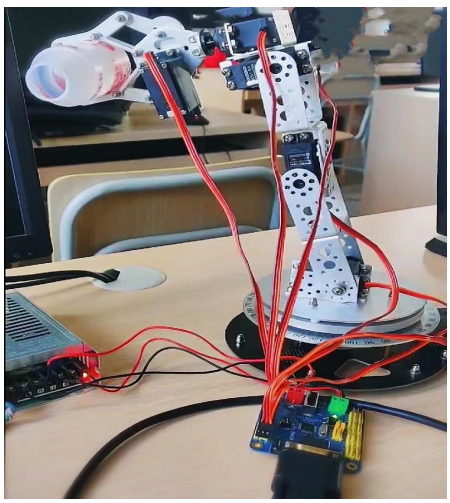
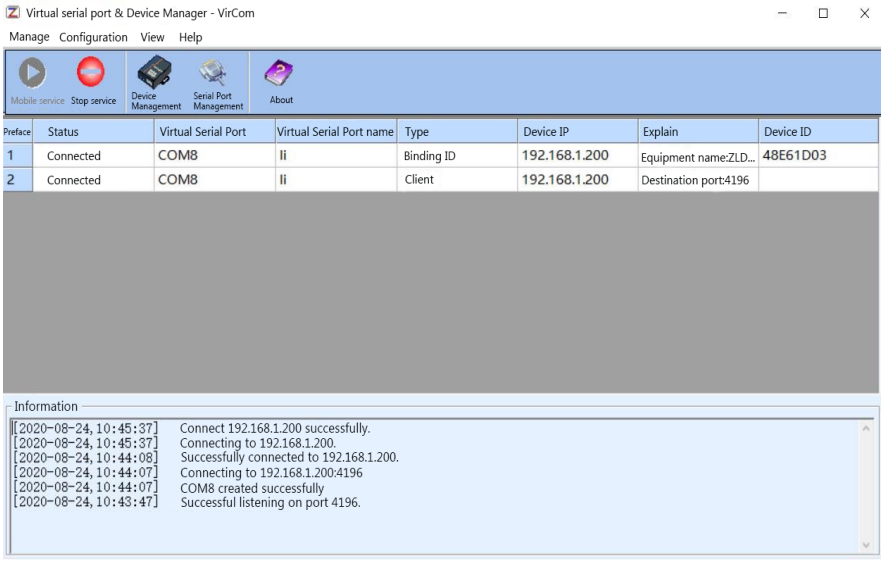
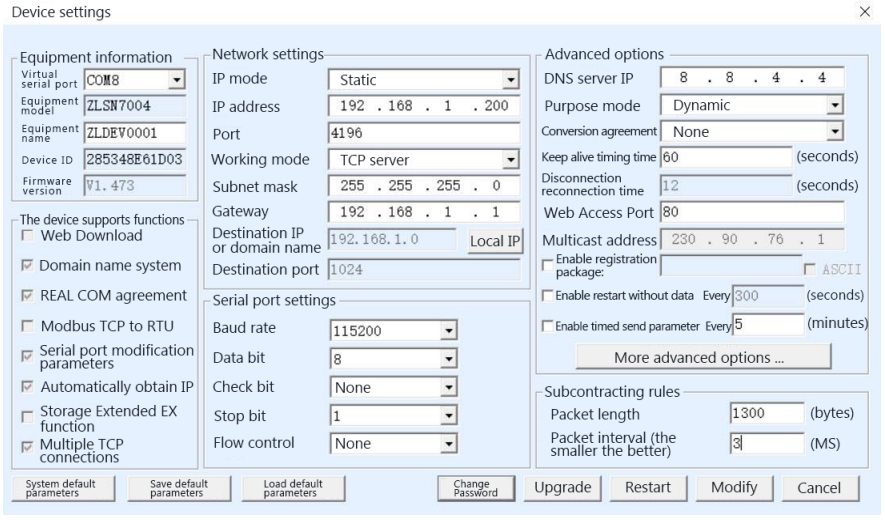


Figure 8. Robotic arm in action.



(a)



(b)

Figure 9. (a) Add virtual serial port COM8; (b) Virtual serial port configuration.

5.2. Remote Control System Implementation

Connect the computer's servo controller software to the configured virtual port COM8 and open the Device Manager to find the virtual port set up, as shown in **Figure 10(a)**. Open the servo servo controller, select the virtual port COM8 and set the baud rate, data bits and stop bits for this port, as shown in **Figure 10(b)**. In the control operation area you can change the PWM signal pulse width by pulling the slider bar corresponding to the servo number, thus realising the control of the servo, as shown in **Figure 10(c)**.

The pulse width value of the control servo can be modified directly above the corresponding servo number. If you want to increase or decrease the robot's running speed, you can enter the corresponding value in the yellow field; the smaller the value, the slower the rotation speed of the servo. In addition, the

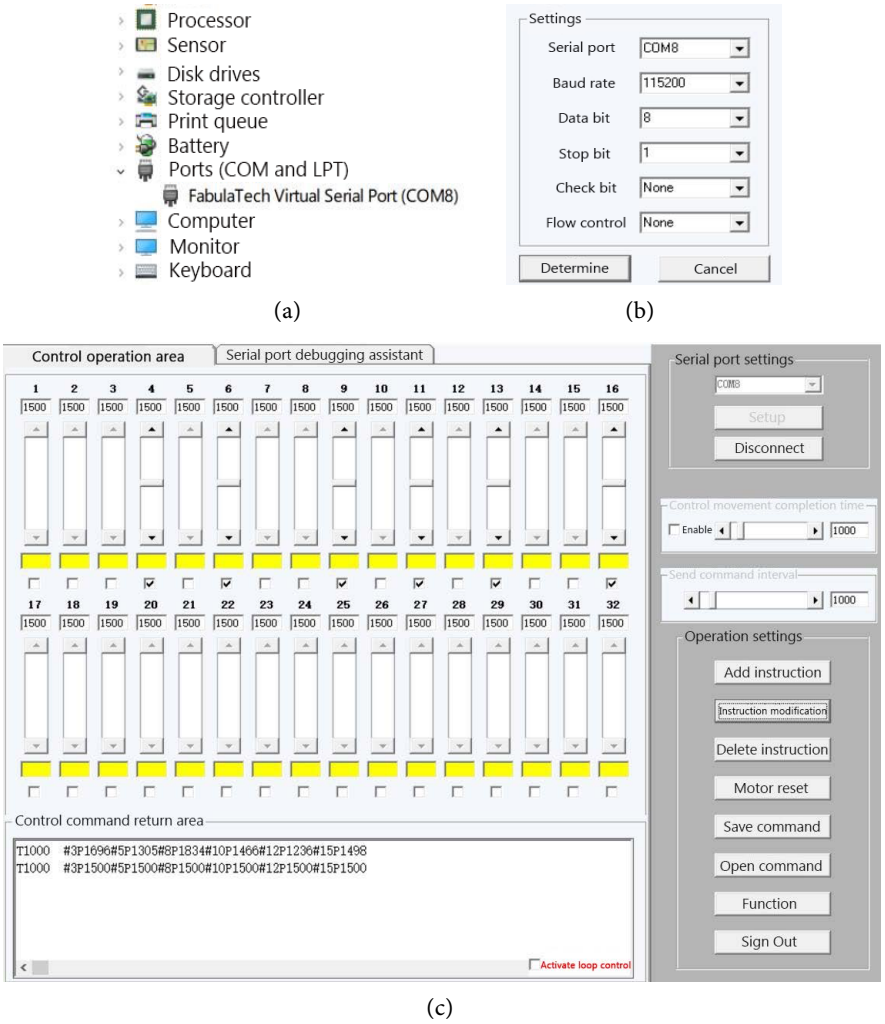


Figure 10. Robotic arm remote control system implementation.

time interval between two commands can be controlled by adjusting the interval between sender action commands.

5.3. DoS Attack on Six-Degree-of-Freedom Robotic Arms

Operating environment: Windows 7 client virtual machine

Attack target: Six-degree-of-freedom robotic arm

Attack target IP: 192.168.1.200

Snort host IP: 192.168.1.103

Simulated intrusion host IP: 192.168.1.150

First, turn on Snort intrusion detection mode before the attack test begin. Then, in the python 3.7 development environment, write a Dos attack on the robotic device, ensuring connectivity to the robotic arm's server and port, and using streaming sockets to establish a connection and send threads to the target port to exhaust its network or system resources. The results of the simulated intrusion into the host attack are shown in **Figure 11(a)**, which shows that the attack target has run out of resources and is down. The robotic arm control system

serial communication appears to be disconnected and cannot continue to connect, as shown in Figure 11(b). Snort detection results show the robotic arm's Dos attack log, as shown in Figure 11(c).

Login to the BASE Analysis Console and check the attack records, you can see that the IP address 192.168.1.200 has recorded a total of 97123 alerts and three different types of attacks, as shown in Figure 12(a). Click Unique Alerts on the Analysis Desk page to see the three types of alarm records. Select the alert tcp alarm type and click on the data below the total directory to see the detailed attack records, including the attack time, source and destination IPs and ports, protocol type and other information, with the record numbers listed in the order of the latest records, as shown in Figure 12(b).

The actual attack test results of the robotic arm control system show that the intrusion detection system based on Snort-BASE can not only actively perform online security detection, but can also perform targeted attack detection through self-designed rules based on the actual situation of the robot control system. It is highly demonstrates the flexibility of this system. At the same time, through the results of visual detection and analysis, each detailed information of the attack can be specifically displayed, so as to facilitate the purposeful deployment of passive defense measures to the control system.

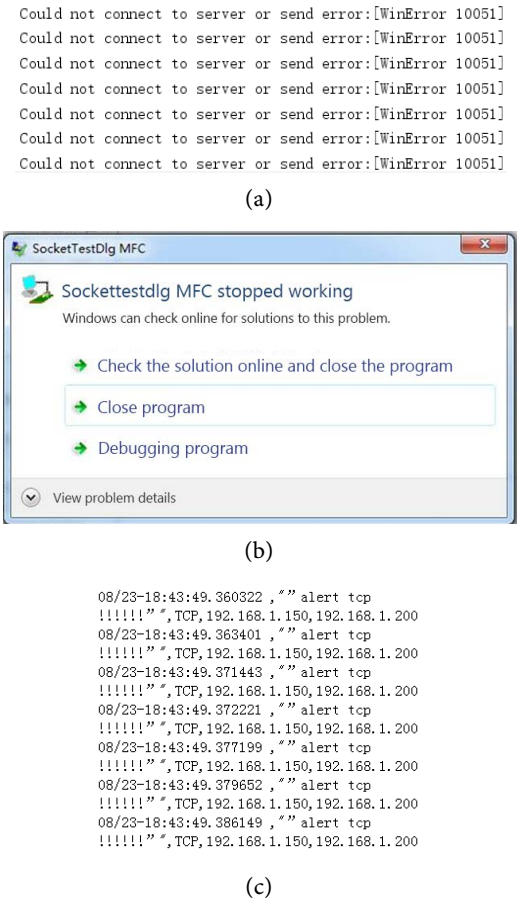


Figure 11. Robotic arm control system attack results.

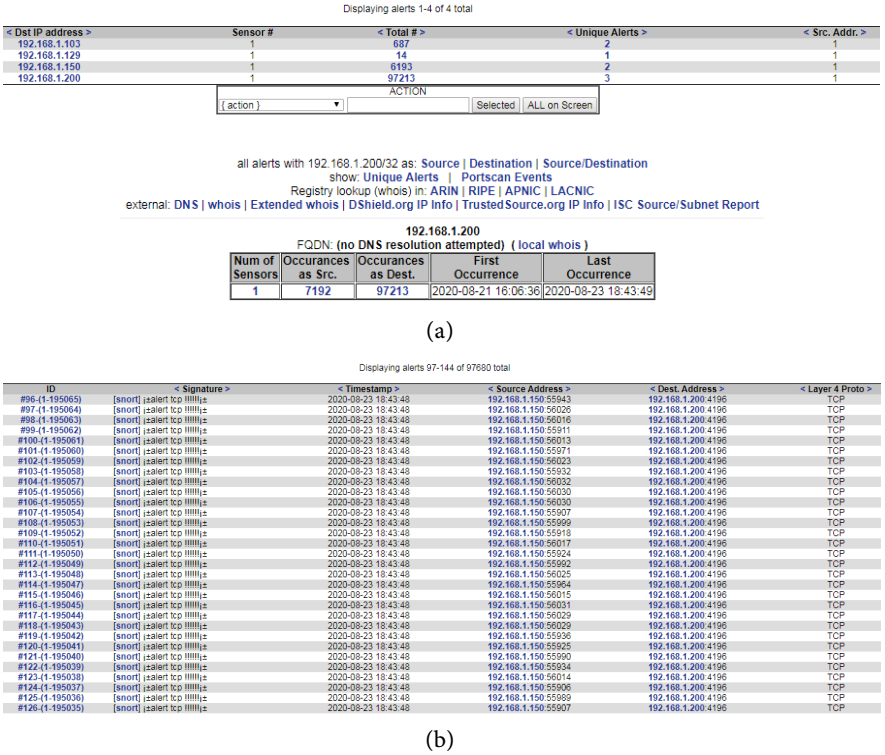


Figure 12. BASE analysis diagram for robotic arm control system attack test.

6. Conclusion

This paper focuses on the detection of communication security for robotic arm control systems, using Snort as the sensor of the detection system and using rules to filter the network traffic collected in real time, and using BASE as the data analyzer of the attack logs, both of which are combined to achieve online real-time detection of robotic arm remote control systems. This paper verifies the effectiveness of the intrusion detection system by designing and implementing a simulated robotic arm control system and building a real six-degree-of-freedom robotic arm control system to achieve active detection of attacks and visualization of attack logs without affecting the real-time requirements of the robot control system.

Fund Projects

General Project of the National Natural Science Foundation of China (No. 61772327); General Project of the Shanghai Natural Science Foundation of China (20ZR1455900); Qi'anxin Big Data Collaborative Security National Engineering Laboratory Open Project (No. QAX-201803); Shanghai Branch Committee Science and Technology Innovation Action Plan (No. 18511105700); Shanghai Science and Technology Commission Electric Power Artificial Intelligence Engineering Technology Research Center Project (NO. 19DZ2252800); Zhejiang University State Key Laboratory of Industrial Control Technology Open Fund (No. ICT1800380).

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Tao, Y., Wang, T.M., Liu, F. and Jiang, S. (2019) Reflections and Suggestions on the Current Situation and Development Trend of Intelligent Robot Research. *High Technology Communication*, 149-163.

[2] Meng, S.N., Liang, Y.B. and Shi, H. (2016) Six-Degree-of-Freedom Robotic Arm Motion Planning Based on ROS Platform. Journal of Shanghai Jiaotong University, **50**, 94-97.

[3] Wu, L.K. (2018) Research on Adaptive Control Method for Large Time Delay Space Remote Operation System. Beijing University of Posts and Telecommunications, Beijing.

[4] Yu, Y.N., Deng, Y.N. and Xie, X. (2021) An Introduction to Common Hacker Attack Methods and Security Protection. *Technology and Innovation*, 74-75+77.

[5] Zhang, Y.H. (2020) Analysis of Cybersecurity Issues of Intelligent Robots. *Journal of Xiangyang Vocational and Technical College*, **19**, 65-68+132.

[6] Cao, K.H., Dong, W.W., Wang, J.L., Zhou, L. and Wang, Y. (2019) An Attack Detection Method for Electricity Consumption Information Collection System Based on Virtual Honeynet. *Computer Science*, **46**, 455-459.

[7] Peng, Y., Jiang, C.Q., Xie, F., *et al.* (2012) Research Progress on Information Security of Industrial Control Systems. *Journal of Tsinghua University* (*Natural Science Edition*), **52**, 1396-1408.

[8] Ding, Y.X., Xiao, M., Liu, A.W. (2009) Research and Implementation on Snort-Based Hybrid Intrusion Detection System. 2009 *International Conference on Machine Learning and Cybernetics*, New York, 2009, 1414-1418. https://doi.org/10.1109/ICMLC.2009.5212282

[9] Khamphakdee, N., Benjamas, N. and Saiyod, S. (2014) Improving Intrusion Detection System Based on Snort Rules for Network Probe Attack Detection. 2014 2*nd International Conference on Information and Communication Technology* (*ICo ICT*), New York, 2014, 69-74. https://doi.org/10.1109/ICoICT.2014.6914042

[10] Liu, H.Y. (2019) Design of Snort-Based Profinet Intrusion Detection System. *Industrial Control Computer*, **32**, 124-127.

Zhang, Y.L., Xie, S.C. and Tang, L.F. (2008) An Efficient Rule Matching Method for Snort Based on Active Rule Sets. *Computer Engineering and Applications*, 124-127.

[11] Gao, L. and Zheng, R.F. (2019) A Network Intrusion Detection and Prevention System Based On Snort and OpenWrt. *Research and Develop*, 17-22+75.

[12] Bai, X.S. (2020) WinPcap-Based Simulation of Stacked Remote Network Microdata Acquisition. *Computer Simulation*, **37**, 333-337.

[13] Chen, J.R. (2018) Research on Snort Preprocessor and Rule Matching Optimization for Web Protection. Changsha University of Technology, Changsha.