A Programmable Routing System for Semi-physical Simulation

Yuan An(✉), Lei Chen, Ping Cui, and Kailiang Zhang

Jiangsu Province Key Laboratory of Intelligent Industry Control Technology, Xuzhou University of Technology, Xuzhou 221018, China
anyuan@xzit.edu.cn

Abstract. The paper designs a programmable routing system, which can be used for semi-physical and network testing. The system adopts the low-cost multi-network card host as the router, and uses the operating system's own security policy to block the connection between the network card and the operating system. The system can prevent the operating system from automatically processing the data frames. The sharpcap is used to capture and send data frames to process network data frames. The transport protocol and routing algorithm are user-defined. During the transmission process, the platform records the changes of routing table and data frame, and comprehensively shows the transmission process of data frame to the user, so that the user can have a more comprehensive understanding of the principle of computer network, and better complete the test and simulation.

Keywords: Routing · Programmable · Computer network · Simulation

1 Introduction

Some new approaches are proposed to improve network routing and measurement [1–3]. Based on analysis on user behavior and traffic, researchers focus on resources utilization [4], energy-efficiency [5] and traffic reconstruction [6]. To test these new technologies, a cheap test platform is essential. Computer network is the product of the fusion of computer and communication technology after a long time of development [7]. Computer network uses communication equipment and lines to connect computer systems with different geographical locations and independent functions [8], so as to improve network software, network communication protocol, information exchange mode and network operating system [9, 10]. The computer network can realize the resource sharing and information transmission in the network better [11]. The local area network(LAN) generally uses a broadcast channel, which can easily access the entire network from a site [12]. The connection between the computer and the external local area network is through a network adapter (Adapter) [13], and the communication between the adapter and the local area network is carried out in a serial transmission mode through cables or twisted pairs [14]. The so-called broadcast communication is when a computer sends data, all computers on the bus can detect this data [15]. In order to realize one-to-one communication on the bus, each computer is equipped with a globally unique MAC address [16]. When sending a data frame, the address of the receiving station is written to the head of the frame. When each computer on the bus detects the data frame, the adapter will match the address of the data frame with its MAC address. If it is consistent, it will be received, otherwise it will be discarded. But in order to manage the network, the adapter has a special working mode: promiscuous mode, in which the adapter will receive all data frames on the bus [17].

The computer network experiment platform generally adopts the sending, capturing and parsing of network data frames to show users the processing process of the data frames by the computer network, and reveals the working principle of the computer network [18–20]. In the existing computer network platform, data frame forwarding and routing information changes are automatically handled by the operating system. Most users can only carry out Socket-based user space network programming and use the network as a transparent transmission channel [21]. Because the IP layer functions and processing mechanisms are mainly reflected in network devices such as routers [22], so it is difficult for users to touch the essence of network technologies, such as IP routing and forwarding during network experiment teaching and network testing, and they cannot intervene in routing information [23, 24]. It is impossible to try custom protocols and algorithms without processing the data frame. Therefore, it restricts the user's understanding of the computer network and limits the teaching and testing effects.

Routers and switches are important components of computer networks, and their internal implementation mechanism is the key to determining the multi-dimensional expansion capabilities of computer networks, such as performance, function, and security [25, 26]. Therefore, this article takes routing design as the entry point of computing network system design and designs a programmable routing system. The system transforms a multi-network card computer into a router, which can process network data frames. The protocol and routing algorithm are defined by the user, and the changes in the routing table and data frame are recorded process. The platform can enhance users' in-depth understanding of computer networks in order to achieve better simulation and testing results.

2 Routing Structure and Implementation Principle

In order to realize the design of the programmable routing system, this research group transform the multi-network card computer into a router to realize the processing of network data frames. The system includes more than two single network card hosts and one multi-network card host, in which the number of network adapters of the multi-network card host is more than the single-network card host. The network adapter of the single network card host and the network adapter of the multi network card host are connected through a network communication device to transfer data frames, and the star network connection is formed between the single network card host and the multi network card host.

The routing structure is divided into a hardware structure and a software structure [3, 27]. The hardware structure extracts data from the bottom layer of the hardware [28]. The software structure is the processing layer, which is responsible for processing the data extracted from the hardware to write its own forwarding strategy [29]. The combination of hardware and software, through the processing of the CPU, to achieve the routing function.

The hardware structure diagram is shown in Fig. 1:

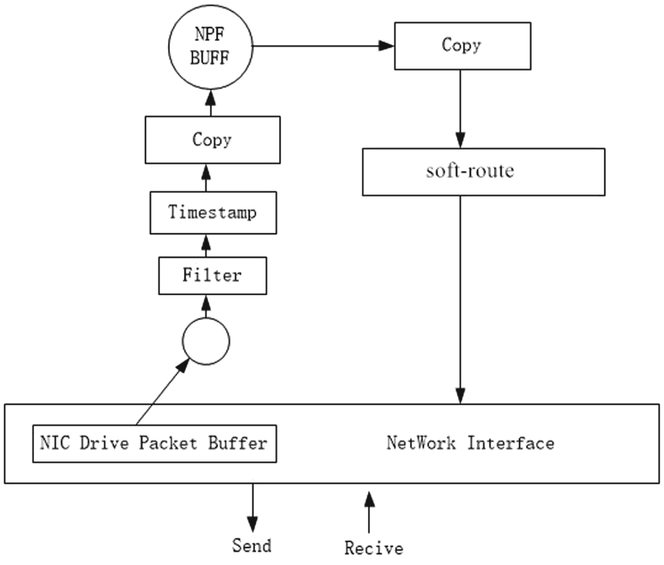


Fig. 1. Routing hardware structure diagram.

NIC Drive Packet Buffer: Network driver, responsible for copying the data on the network card.

Filter: Filter the data captured by the user at the kernel layer.

Timestamp: The time interval returned after each acquisition.

Copy: Copy the data of the kernel layer to the user memory.

NPF BUFF: Parallel and serial buffer.

NPF BUFF Copy: Take out the data and put it into the soft route for processing. The processed structure is sent directly from the network card through the soft route.

The software structure is shown in Fig. 2:

Packet_callback: routing function for soft routing to process packets.

DLL extension: the interface reserved for users to use for processing.

Ordinary PC computers only have the function of local routing, that is, they can only forward their own IP data packets, and cannot forward any other data packets [30, 31]. The core of the realization of soft routing is that compared with commercial routers on the market, the forwarding process and the principle of forwarding must be considered [32, 33]. When soft routing receives data from a certain network card, it first decrements the TTL time of its IP packet header. If the TTL time is found to be 0, the data packet is discarded, otherwise the routing table encapsulates it and replaces the Ethernet header. At the same time, the soft route checks the destination address and performs an AND operation with the destination address of the routing table [34, 35]. The longest match

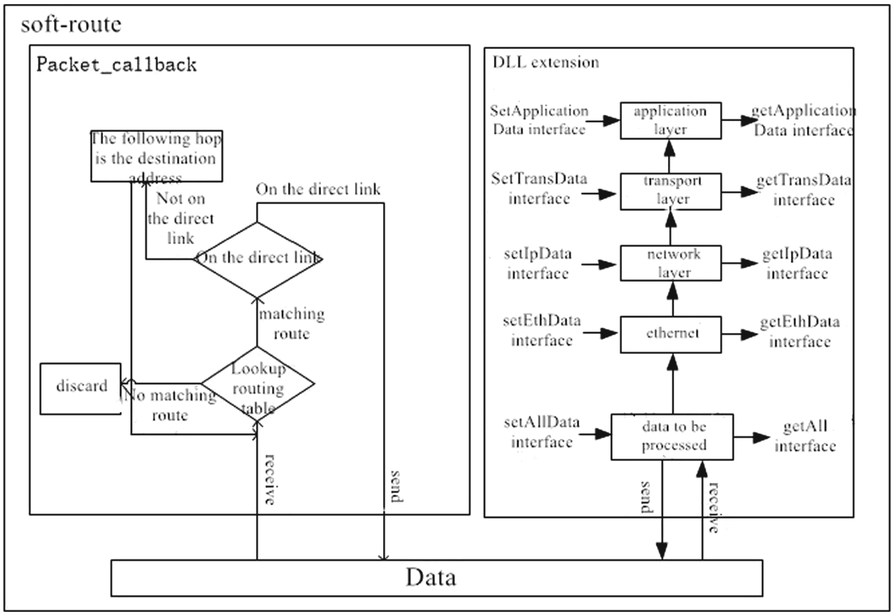


Fig. 2. Software structure diagram.

is the route that matches the forwarding, and according to the packet encapsulation, the checksum of the IP packet is recalculated and sent out from the network interface. Capturing and sending packets are implemented using winpcap. The route forwarding process is shown in Fig. 3.

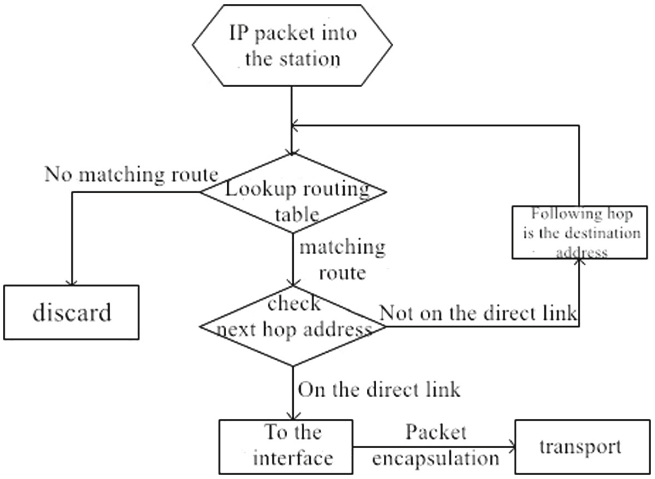


Fig. 3. Routing and forwarding flowchart.

3 Implementation Process

The programmable routing system designed in this paper can be used for computer network experiment teaching and network testing. Users can design protocols and routing algorithms by themselves and observe the routing process to deepen their understanding of the working principle of computer networks.

According to the system implementation technical solution, the system architecture diagram is shown in Fig. 4.

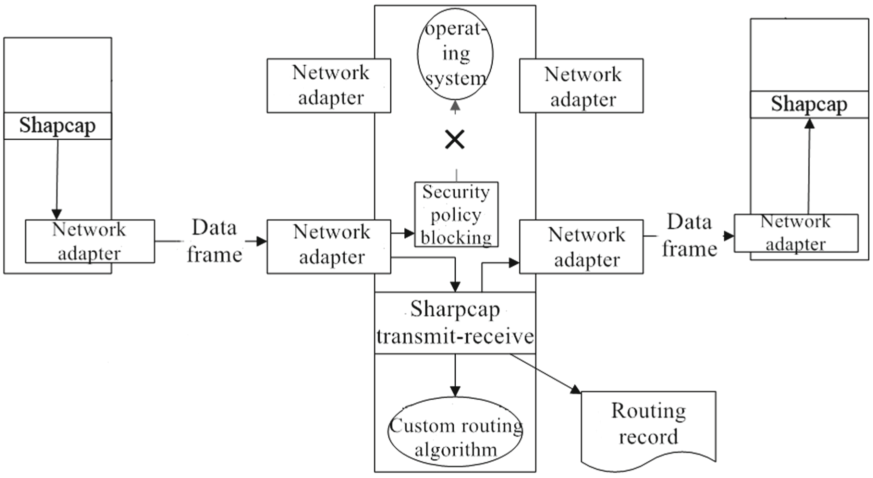


Fig. 4. System architecture diagram.

It can be seen from Fig. 4 that the system includes more than two single network card hosts and one multi-network card host. The number of network adapters of the multi-network card host is greater than that of the single-network card host. The network adapter of the single network card host and the network adapter of the multi-network card host are connected through a network communication device to transfer data frames. A single network card host and a multi-network card host form a star connection. A single network card host is provided with a sharpcap interface for information exchange with its internal network adapter. It is blocked by the security policy between the network adapter of the multi-NIC host and its own operating system. The sharpcap interface is set in the multi-NIC host, and the packet capture, transmission and processing are carried out through the sharpcap interface between the multiple network adapters in the multi-NIC host. During the transmission of data packets between the network adapters of multiple network card hosts, the data packets are captured and processed according to the custom routing algorithm, and the transmitted data information is recorded at the same time.

The programmable routing system uses sharpcap and winpcap to capture and send network data frames, and uses security policies to prevent the operating system from accepting data frames. The experimental software processes the data frames, thereby transforming the multi-network card computer into a router. Users can design protocols and write their own routing algorithms freely. The system can record the changes of routing tables and data frames, and show the transmission process of data frames to user.

The sharpcap interface in this system is a direct interface for monitoring software to call Winpcap. It has the functions of packet capture, injection, analysis and construction to form a packet capture framework [29]. This framework is a perfect combination of Winpcap components and Windows network core functions. The framework set inherits

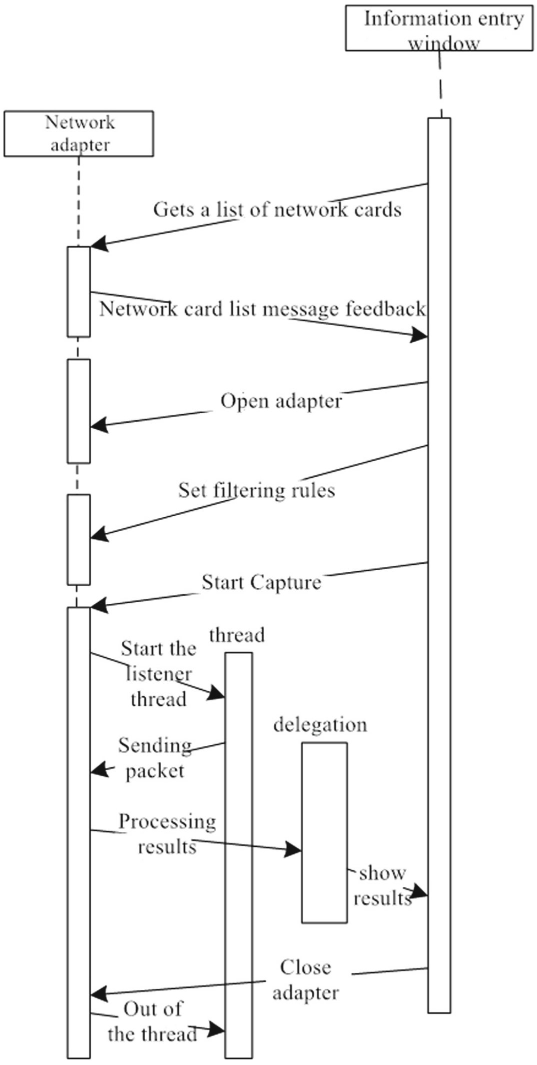


Fig. 5. System implementation process diagram.

Winpcap and surpasses Winpcap. It uses a message mechanism to maximize the superiority of object-oriented performance [36]. It also integrates some network API functions and API functions for reading the registry [37].

The process of the system using Sharpcap to capture packets is shown in Fig. 5. First, get the adapter list, get the MAC address and gateway address of the adapter, then open the corresponding adapter according to the operation selection, and then specify the corresponding filtering criteriafor the adapter, and start capturing information passed by the adapter. While capturing, you need to receive the captured data packets, process the data, and display the data, and finally close the adapter. In the process of data packet reception, the design of the reception affects the efficiency and accuracy of this capture, and even determines the success or failure of the captured data packet. During the capture process, sharpcap needs to start a new thread in the host memory, which is specifically responsible for listening to the adapter for the arrival of data packets. The meaning of this thread is to be separated from the main form thread, to avoid the main thread from blocking when a packet arrives, and to prevent the program from "dead".

The security strategy adopted in the system refers to the access protocol for the operating system to process the data packets in the network adapter. Security policy blocking refers to using the operating system's own security policy to block the connection between the network card and the operating system, preventing the operating system from automatically processing data frames.

The flow chart of the system using sharpcap for data packet capture is shown in Fig. 6:

The specific implementation steps of the above programmable routing platform are as follows:

1) Edit the link layer data frame

The data link layer is responsible for combining bits into bytes and combining bytes into frames. Frames are used at the data link layer, and the data packets passed from the network layer are encapsulated into frames for transmission according to the type of media access. The 802.3 frame is encapsulated on the network adapter, and its structure includes the preamble, destination address, original address, length, data and frame check sequence. The learner uses the interface software to edit each part of the frame as needed, and writes it into the encapsulation package to form a data package to be used.

2) Use sharpcap to send the edited data frame on the single NIC host side

The core technology in the network adapter is the random contention-based media access method, that is, the CSMA/CD method. The process of sending the edited data frame using sharpcap is as follows: a. Carrier sense, because Ethernet data uses Manchester side coding, it can determine whether the bus level jumps to determine whether the bus is idle; b. Conflict detection, both hosts hear that the bus is idle, both will send data, there may be conflicts, so conflict detection should be performed, and conflicts should stop sending data; Random delay retransmission, after stopping, the node performs a random delay retransmission. If the retransmission is still unsuccessful after 16 times, the transmission failure is declared.

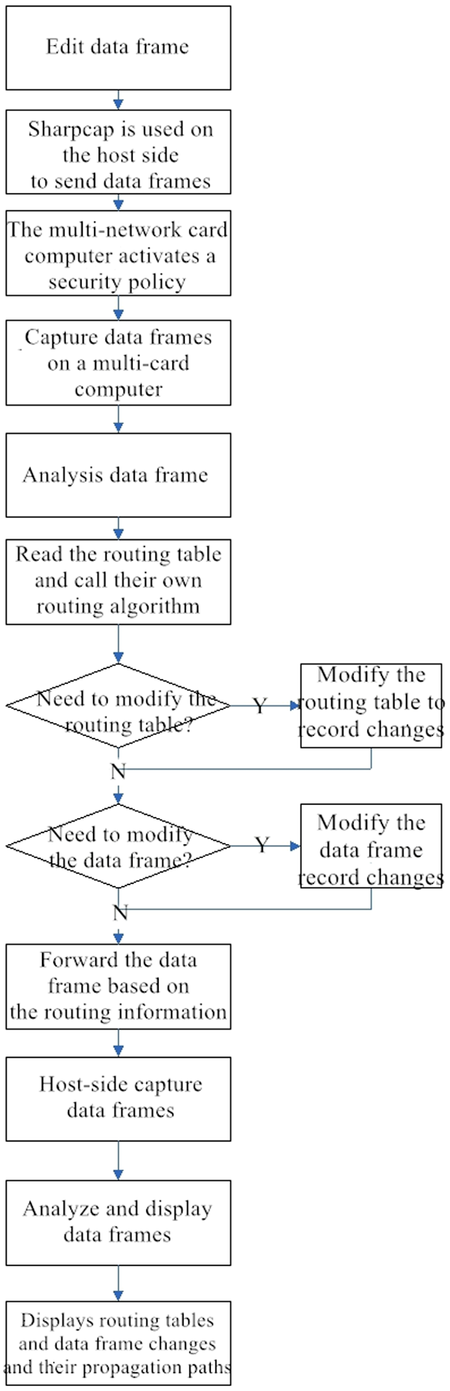

Edit data frame
Sharpcap is used on the host side to send data frames
The multi-network card computer activates a security policy
Capture data frames on a multi-card computer
Analysis data frame
Read the routing table and call their own routing algorithm
Need to modify the routing table?
Y
Modify the routing table to record changes
N
Need to modify the data frame?
Y
Modify the data frame record changes
N
Forward the data frame based on the routing information
Host-side capture data frames
Analyze and display data frames
Displays routing tables and data frame changes and their propagation paths

Fig. 6. Flow chart of packet capture using sharpcap.

3) Use multiple network card hosts as routers to receive data frames

In the process of receiving data frames mainly using network adapters, first check whether there is a collision or whether a frame is dropped, otherwise go to the next step; check the destination address of the frame to see if the frame can be accepted, if it can, then go to the next step; Check the CRC checksum and LLC data length. If both are correct, accept the frame, otherwise discard it.

4) Using security policies to prevent the operating system from automatically processing data frames

The use of the operating system's own security strategy blocks the connection between the network card and the operating system, and prevents the operating system from automatically processing the data frames. This prevents the loss of data frames and provides important protection for subsequent users.

5) Capture data frames on multiple network card hosts and analyze the data frames

In the packet parsing thread, it is necessary to analyze the content of the actual packet. The packet data captured by Sharpcap is stored in an instance of the RowPacket class. This instance provides a byte-type indexer to read the actual packet data. The indexer can be used as a byte-type read-only array. The packet information is stored in bytes. The data packet is generally a standard Ethernet frame, that is, the first 14 bits are MAC information; if it is a TCP/IP protocol frame, the 15–34 bytes are IP information, and the 35–55 bytes are TCP information; If it further connotes HTTP data packets, the bytes that follow are HTTP request and response information.

6) Read the routing table and call to write routing rules

According to the requirements of capturing data frames, the user writes the corresponding routing rules and stores them in the form of routing tables, and finally captures the corresponding data frames through custom routing rules.

7) Modify the data frame and routing table according to the algorithm;

After reading the captured data frame, it can be modified to make it familiar to the user to manipulate the format of the data frame and the construction of the routing table.

8) Forward and record data frame and routing table changes;
9) The host side of the single network card captures the message and identifies the data frame;

Data packet capture is used to intercept the data packets sent by users in the LAN at the transport layer and put the captured data packets into the program buffer; After receiving the captured data packet, the packet analysis module reads the data packet from the buffer, filters the junk data packet through the set filtering rules, and analyzes the packet after decomposing it to analyze the packet and improve The efficiency of capturing and parsing data packets; The decoding module automatically selects the appropriate decoding method to decode the search string according to the encoding rules, and is used to restore the search data string encoded by the browser.

10) Display data frame changes and propagation path, routing table changes

The display data frame is the analysis result of the data packet of the user's retrieval information in the current local area network, including the capture time of the data packet, source IP, source MAC, destination IP, search engine host name, use protocol, search string, original message and other related information, display data frame changes and propagation path, routing table changes.

4 Experimental Test

The router in the system is a special PC, which is equivalent to a forwarding station. When two PCs are communicating, it can decide whether to forward the data packet based on whether the destination network address is in the destination network segment.

Both hardware routers and software routers are classified as routers. The hardware devices in the hardware router are specially designed for the router to forward data. The software router does not have special hardware to handle forwarding. It uses a CPU, which can be said to be a process of the user's PC.

The "Routing Remote Access" service is manually turned on for the user. When it is turned on, it indicates that this PC can perform routing and forwarding, and when it is turned off, it indicates that this PC cannot perform routing and forwarding.

The steps of configuring routing and remote access in the experimental test are as follows:

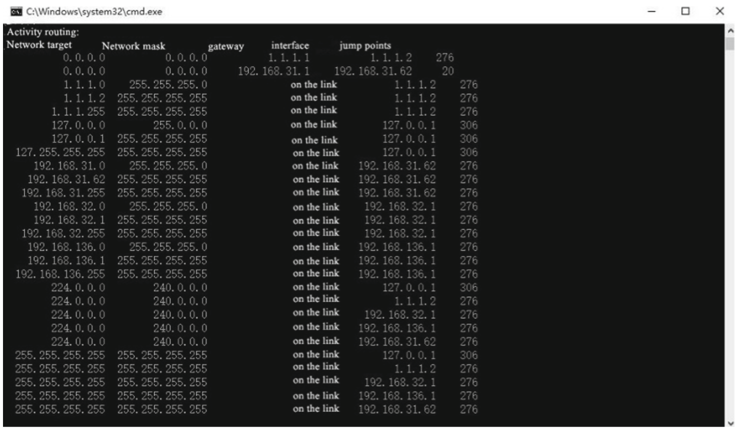
Click "Administrative Tools" in "Start", and select "Routing and Remote Access" function in the administrative tools, then you will see the configuration wizard; select the local server, click "Operation"-"Configure and enable routing and remote access"; click "Next", select "Custom Configuration"-"Next"; select "LAN Routing" and then click "Next" and finally click Finish to start routing and remote access.

Create a new batch of processing file myroute.bat, and place it in the directory folder that starts automatically at boot, and edit the content of myroute.bat as follows:

```
Route add 192.168.1.0 mask 255.255.255.0 192.168.32.1
```

Double-click the bat batch to change the routing table.

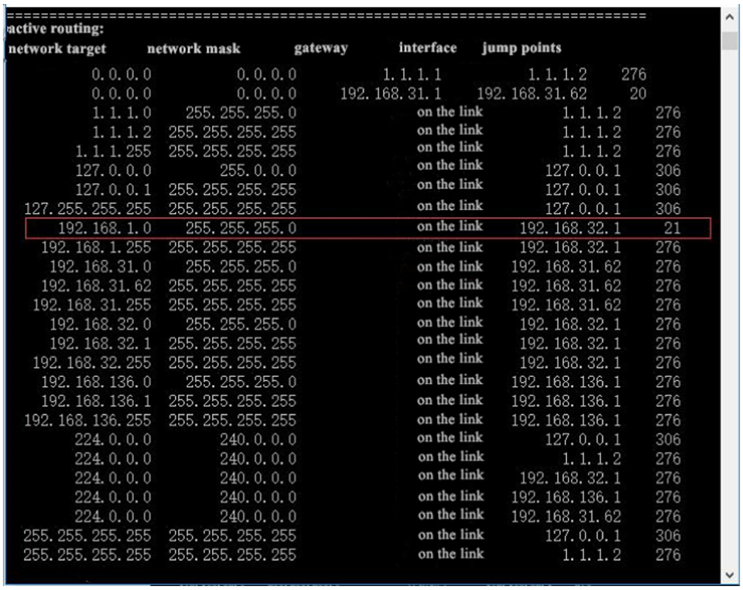
The routing table before running is shown in Fig. 7:



```
C:\Windows\system32\cmd.exe
Activity routing:
Network target      Network mask        gateway      interface      jump points
          0.0.0.0          0.0.0.0          1.1.1.1          1.1.1.2    276
          0.0.0.0          0.0.0.0     192.168.31.1     192.168.31.62     20
          1.1.1.0    255.255.255.0      on the link          1.1.1.2    276
          1.1.1.2  255.255.255.255      on the link          1.1.1.2    276
        1.1.1.255  255.255.255.255      on the link          1.1.1.2    276
        127.0.0.0        255.0.0.0      on the link        127.0.0.1    306
        127.0.0.1  255.255.255.255      on the link        127.0.0.1    306
  127.255.255.255  255.255.255.255      on the link        127.0.0.1    306
     192.168.31.0    255.255.255.0      on the link    192.168.31.62    276
    192.168.31.62  255.255.255.255      on the link    192.168.31.62    276
   192.168.31.255  255.255.255.255      on the link    192.168.31.62    276
     192.168.32.0    255.255.255.0      on the link     192.168.32.1    276
     192.168.32.1  255.255.255.255      on the link     192.168.32.1    276
   192.168.32.255  255.255.255.255      on the link     192.168.32.1    276
    192.168.136.0    255.255.255.0      on the link    192.168.136.1    276
    192.168.136.1  255.255.255.255      on the link    192.168.136.1    276
  192.168.136.255  255.255.255.255      on the link    192.168.136.1    276
        224.0.0.0        240.0.0.0      on the link        127.0.0.1    306
        224.0.0.0        240.0.0.0      on the link          1.1.1.2    276
        224.0.0.0        240.0.0.0      on the link     192.168.32.1    276
        224.0.0.0        240.0.0.0      on the link    192.168.136.1    276
        224.0.0.0        240.0.0.0      on the link    192.168.31.62    276
  255.255.255.255  255.255.255.255      on the link        127.0.0.1    306
  255.255.255.255  255.255.255.255      on the link          1.1.1.2    276
  255.255.255.255  255.255.255.255      on the link     192.168.32.1    276
  255.255.255.255  255.255.255.255      on the link    192.168.136.1    276
  255.255.255.255  255.255.255.255      on the link    192.168.31.62    276
```

Fig. 7. Before adding the local routing table.

After adding the route, the routing table is shown in Fig. 8 below:



```
===========================================================================
active routing:
network target      network mask        gateway      interface      jump points
          0.0.0.0          0.0.0.0          1.1.1.1          1.1.1.2    276
          0.0.0.0          0.0.0.0     192.168.31.1     192.168.31.62     20
          1.1.1.0    255.255.255.0      on the link          1.1.1.2    276
          1.1.1.2  255.255.255.255      on the link          1.1.1.2    276
        1.1.1.255  255.255.255.255      on the link          1.1.1.2    276
        127.0.0.0        255.0.0.0      on the link        127.0.0.1    306
        127.0.0.1  255.255.255.255      on the link        127.0.0.1    306
  127.255.255.255  255.255.255.255      on the link        127.0.0.1    306
      192.168.1.0    255.255.255.0      on the link     192.168.32.1     21
    192.168.1.255  255.255.255.255      on the link     192.168.32.1    276
     192.168.31.0    255.255.255.0      on the link    192.168.31.62    276
    192.168.31.62  255.255.255.255      on the link    192.168.31.62    276
   192.168.31.255  255.255.255.255      on the link    192.168.31.62    276
     192.168.32.0    255.255.255.0      on the link     192.168.32.1    276
     192.168.32.1  255.255.255.255      on the link     192.168.32.1    276
   192.168.32.255  255.255.255.255      on the link     192.168.32.1    276
    192.168.136.0    255.255.255.0      on the link    192.168.136.1    276
    192.168.136.1  255.255.255.255      on the link    192.168.136.1    276
  192.168.136.255  255.255.255.255      on the link    192.168.136.1    276
        224.0.0.0        240.0.0.0      on the link        127.0.0.1    306
        224.0.0.0        240.0.0.0      on the link          1.1.1.2    276
        224.0.0.0        240.0.0.0      on the link     192.168.32.1    276
        224.0.0.0        240.0.0.0      on the link    192.168.136.1    276
        224.0.0.0        240.0.0.0      on the link    192.168.31.62    276
  255.255.255.255  255.255.255.255      on the link        127.0.0.1    306
  255.255.255.255  255.255.255.255      on the link          1.1.1.2    276
```

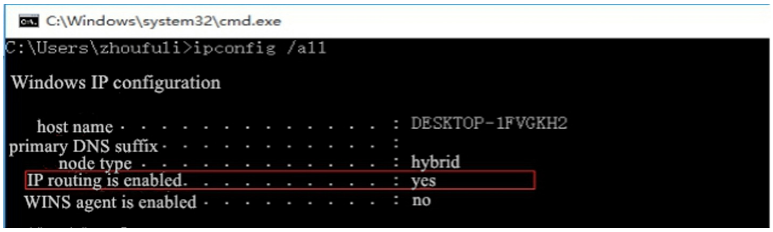
Fig. 8. After adding the local routing table.

Can complete the 192.168.1.0/24 destination network forwarding.
WIN7, WIN10 open routing and remote access modification

1) Open the registry editor.
2) Open the registry editor and find the following entries:

HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Services/Tcpip/Parameters.
Select the following item: IPEnableRouter: REG_DWORD: 0×0.

To enable IP forwarding for all network connections installed and used on this computer, the value is 1 and off is 0. After the modification is complete, enter ipconfig/all on the CMD command line to view the opening information as shown in Fig. 9 below:



```
C:\Windows\system32\cmd.exe
C:\Users\zhoufuli>ipconfig /all

Windows IP configuration

   host name . . . . . . . . . . . . . . : DESKTOP-1FVGKH2
primary DNS suffix . . . . . . . . . . . :
     node type . . . . . . . . . . . . . : hybrid
  IP routing is enabled. . . . . . . . . : yes
  WINS agent is enabled . . . . . . . . : no
```

Fig. 9. NIC information.

In order to enable a process to call code that does not belong to it, we need a way to call it. Dynamic linking provides a way to make the process dynamically load and execute. The code of the function is encapsulated in a DLL file, the DLL contains one or more executable functions. And DLL also helps to share data and resources. When multiple DLLs are loaded, the shared resources can be accessed and modified at the same time.

When a program needs to be expanded, in order not to affect the operation of its normal modules, the method of dynamic link library can be used to make it easier and simpler to complete the expansion of the program. The system uses the method of displaying and calling the dynamic link library. Nothing needs to be operated during compilation. When the functions in the DLL need to be used, the DLL can be loaded and used through the two API functions LoadLibrary () and FindProcAdress ().

5 Conclusion

This article details the design and implementation of the programmable routing system. The system uses a low-cost multi-network card host as a router, and uses the operating system's own security strategy to block the connection between the network card and the operating system, preventing the operating system from automatically data frames deal with. The system simultaneously uses sharpcap to capture and send data frames. Experimental test results show that the system can realize functions such as routing and forwarding, remote access, recording data frames and routing table information.

Therefore, experimenters can use the system interface to design routing strategies and network protocols, and use the system to record information to understand the routing process. Therefore, the system helps experimenters to deepen their understanding of the working principle of computer networks, and better complete semi-physical simulation work.

Acknowledgements. This work is partly supported by Jiangsu technology project of Housing and Urban-Rural Development (No.2018ZD265, No.2019ZD039, No.2019ZD040, No.2019ZD041).

References

1. Zhang, K., Chen, L., An, Y., et al.: A QoE test system for vehicular voice cloud services. Mobile Network Application (2019). https://doi.org/10.1007/s11036-019-01415-3
2. Wang, F., Jiang, D., Qi, S.: An adaptive routing algorithm for integrated information networks. Chin. Commun. **7**(1), 196–207 (2019)
3. Hu, G., Xu, K., et al.: Research and development on programmable router. Chin. Educ. Netw. **65**(7), 40–43 (2010)
4. Huo, L., Jiang, D., Zhu, X., et al.: An SDN-based fine-grained measurement and modeling approach to vehicular communication network traffic. International Journal of Communication Systems, Online Available, pp. 1–12 (2019)
5. Huo, L., Jiang, D., Lv, Z., et al.: An intelligent optimization-based traffic information acquirement approach to software-defined networking. Computational Intelligence, pp. 1–21 (2019)
6. Chen, L., Jiang, D., Bao, R., Xiong, J., Liu, F., Bei, L.: MIMO scheduling effectiveness analysis for bursty data service from view of QoE. Chin. J. Electron. **26**(5), 1079–1085 (2017)
7. Lu, G., Shi, Y., Guo, C., et al.: A configurable packet forwarding engine for data center networks. In: Proceedings of the 2nd Workshop on ACM SIGCOMM, pp. 25–30 (2009)
8. Sommers, J., Barford, P., Cmvella, M.: Router primitives for programmable active measurement. In: The 2nd Workshop on ACM SIGCOMM, pp. 13–18 (2009)
9. Jiang, D., Wang, Y., Lv, Z., et al.: Big data analysis-based network behavior insight of cellular networks for industry 4.0 applications. IEEE Trans. Ind. Inform. **16**(2), 1310–1320 (2020)
10. Bolla, R., Bmsch, R.: PC-based software routers:high performance and application service support. In: The 1st Workshop on ACM SIGCOMM, pp. 27–32 (2008)
11. Spalink, T., Kariin, S., Peterson, L., et a1.: Building a robust software based muter using network processors. In: The 18th ACM Symposium on Operating Systems Principles(SOSP), pp. 216–229 (2001)
12. Jiang, D., Huo, L., Song, H.: Rethinking behaviors and activities of base stations in mobile cellular networks based on big data analysis. IEEE Trans. Netw. Sci. Eng. **1**(1), 1–12 (2018)
13. Chen, L., et al.: A lightweight end-side user experience data collection system for quality evaluation of multimedia communications. IEEE Access **6**(1), 15408–15419 (2018)
14. Jiang, D., Huo, L., Li, Y.: Fine-granularity inference and estimations to network traffic for SDN. Plos One **13**(5), 1–23 (2018)
15. Chen, L., Zhang, L.: Spectral efficiency analysis for massive MIMO system under QoS constraint: an effective capacity perspective. Mobile Network Application (2020). https://doi.org/10.1007/s11036-019-01414-4
16. Wang, F., Jiang, D., Qi, S., et al.: A dynamic resource scheduling scheme in edge computing satellite networks. Mobile Networks and Applications, Online Available (2019)

17. Yu, M., Thottan, M., Li, L.: Latency equalization: a programmable muting service primitive. In: The 1st Workshop on ACM SIGCOMM, pp. 39–44 (2008)
18. Lam, C.F.: Passive Optical Networks: Principles and Practice. Academic Press, San Diego (2007)
19. Jiang, D., Wang, Y., Lv, Z., et al.: A joint multi-criteria utility-based network selection approach for vehicle-to-infrastructure networking. IEEE Trans. Intell. Transp. Syst. **19**(10), 3305–3319 (2018)
20. Jiang, D., Zhang, P., Lv, Z., et al.: Energy-efficient multi-constraint routing algorithm with load balancing for smart city applications. IEEE Internet Things J. **3**(6), 1437–1447 (2016)
21. Yang, Y., Yang, J., Qin, D.: Data center network multi-path routing algorithm. J. Tsinghua Univ. (Nat. Sci. Edn.) **3**, 262–268 (2016)
22. Jorgensen, B.: Guide to Network Programming (2008)
23. Jiang, D., Wang, Y., Lv, Z., Wang, W., Wang, H.: An energy-efficient networking approach in cloud services for IIoT networks. IEEE J. Sel. Areas Commun. **38**(5), 928–941 (2020)
24. Ibrahim, A., Alfa, A.: Optimization techniques for routing design problems over wireless sensor networks: a short tutorial. In: SENSORNETS 2017 - Proceedings of the 6th International Conference on Sensor Networks, pp. 156–167 (2017)
25. Jiang, D., Wang, Y., Lv, Z., Qi, S., Singh, S.: Big data analysis based network behavior insight of cellular networks for industry 4.0 applications. IEEE Trans. Ind. Inform. **16**(2), 1310–1320 (2020)
26. Huo, L., Jiang, D., Qi, S., et al.: An AI-based adaptive cognitive modeling and measurement method of network traffic for EIS. Mobile Networks and Applications, Online Available (2019)
27. Zhang, X., Liu, Z., Zhao, Y., et al.: Scalable router. J. Softw. **19**(6), 1452–1464 (2008)
28. Jiang, D., Li, W., Lv, H.: An energy-efficient cooperative multicast routing in multi-hop wireless networks for smart medical applications. Neurocomputing **220**(2017), 160–169 (2017)
29. Wu, D., Yin, Y.F., Lawphongpanich, S., Yang, H.: Design of more equitable congestion pricing and tradable credit schemes for multimodal transportation networks. Transp. Res. Part B Methodol. **46**(9), 1273–1287 (2012)
30. Jiang, D., Wang, Y., Lv, Z., et al.: Intelligent optimization-based reliable energy-efficient networking in cloud services for IIoT networks. IEEE Journal on Selected Areas in Communications, Online Available (2019)
31. Zhang, L.H., Yang, H., Wu, D., Wang, D.H.: Solving a discrete multimodal transportation network design problem. Transp. Res. Part C Emerg. Technol. **49**, 73–86 (2014)
32. Qi, S., Jiang, D., Huo, L.: A prediction approach to end-to-end traffic in space information networks. Mobile Networks and Applications, Online Available (2019)
33. Jiang, D., Wang, W., Shi, L., et al.: A compressive sensing-based approach to end-to-end network traffic reconstruction. IEEE Trans. Netw. Sci. Eng. **5**(3), 1–12 (2018)
34. Wang, Y., Jiang, D., Huo, L., et al.: A new traffic prediction algorithm to software defined networking. Mobile Networks and Applications, Online Available (2019)
35. Cheng, P., Xu, C.W., Lebreton, P., Yang, Z.D., Chen, J.M.: TERP: time-event-dependent route planning in stochastic multimodal transportation networks with bike sharing system. IEEE Internet of Things J. **6**(3), 4991–5000 (2019)
36. Peng, Y., Wang, W.Y., Guo, Z.J., Song, X.Q., Zhang, Q.: A stochastic seaport network retrofit management problem considering shipping routing design. Ocean Coast. Manag. **119**, 169–176 (2016)
37. Murata, H.: A new routing design methodology for multi-chip IC packages. Midwest symposium on circuits and systems. In: The 2004 47th Midwest Symposium on Circuits and Systems - Conference Procceedings, pp. 1473–1476 (2004)