A Method for Realizing Covert Communication at Router Driving Layer

Jingsong Cui[1], Chi Guo[2(✉)], Manli Zhang[3], and Qi Guo[3]

[1] School of Computer Science, Wuhan University, Wuhan 430079, China
[2] GNSS Research Center, Wuhan University, Wuhan 430079, China
guochi@whu.edu.cn
[3] School of Cyber Science and Engineering, Wuhan University, Wuhan 430079, China

Abstract. The existing information hiding methods mainly focus on the analysis of the header field of the network protocol and the researches of VoIP. Well, the location of embedded covert data is easy to detect, its capacity is limited and the condition of covert communication is limited. In this paper, we propose a method which builds a covert channel between two routers for transmitting large-capacity information at the driver layer. The router is divided into sender and receiver, both of which mount our own driver and user application, intercept UDP packets generated during the user's voice or video call with instant message software. We analyze the UDP characteristics and construct UDP meta-model, and then split the secret information into the payload part of the meta-model with CRC check. The forged UDP is sent out with the common UDP traffic. The receiver router intercepts and identifies the forged UDP packets by CRC check and utilizes the obtained forged UDP to restore the original information. Moreover, we exploited WeChat and QQ voice call to conduct numerous simulations of covert communication, and successfully transmitted the secret information transparently from a network-restricted area to a more relaxed area of network supervision, verifying the concealment of the method.

Keywords: UDP packets · Covert channel · Instant message software · Router driver layer

1 Introduction

With the popularity of the Internet, there have been many security problems. How to ensure the security of data transmission over the Internet is an issue that needs to be urgently addressed. The existing methods for protecting data transmission are shown in Fig. 1, roughly including four fields: encrypt the network data, hide data in the static carrier, embed data in the redundant fields of network protocol, and embed data in the payload of voice packet. In detail, the mechanism of traditional information protection is to encrypt network communication. However, the encrypted data is disorganized and unusual, which is prone to cause the attacker's suspicions and will be attacked and destroyed. In addition, the static carrier such as image, text is not secure either, which can leave evidence for detection and is not real-time. Information hiding technology is

W. Han et al. (Eds.): CTCIS 2019, CCIS 1149, pp. 104–118, 2020.
https://doi.org/10.1007/978-981-15-3418-8_8

an emerging technology. As one of information hiding, the covert channel was first proposed by Lampson [1]. He viewed covert communication as a process of communicating data, through a transferring channel that is neither designed nor intended. A covert channel is defined as a communication channel where information is transferred in a manner that violates the system security policy. As for covert channel, there are mainly two major areas. In the beginning, network protocols are popular used as carriers, where secret information can be embedded in fields such as redundant fields, fuzzy fields, extension bits, and padding bits for covert communication [2]. The fields that are often used to embed secret information include the ID of IP protocol, TTL, the ISN of TCP protocol, the time-stamp portion of the options field in the ICMP echo request, and so on. Because these fields are short in length (within 2 bytes), the secret information that can be embedded therein is limited. Moreover, the current covert channel detection can detect the hidden channel established by the network protocol, so the channel is not secure. Only a few researchers construct a covert channel through the payload of the audio packet (Lost Audio Packet [3]), but it has the risk of lost packets. And then more and more research tends to utilize the above methods to hide the secret information in VoIP streams, just as Fig. 1 shows. However, the effect of the covert channel based on VoIP voice stream depends on the specific speech coding format, and capacity is not very large.

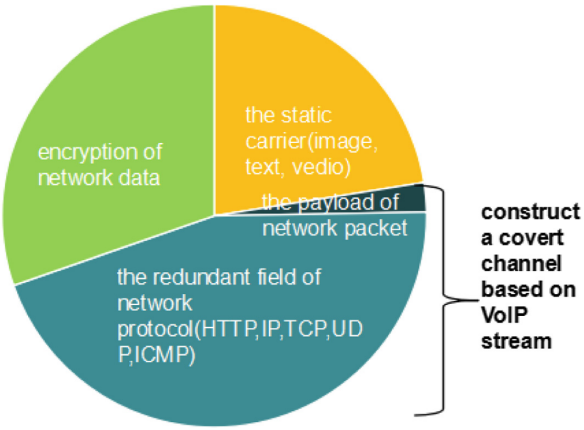

Fig. 1. Methods of protecting data transmission

The shortages in existing works motivate us to design a novel method. In this paper, we propose a method for realizing covert communication at router driving layer. The contributions of our work can be summarized as follows:

- We become the first one to realize a covert channel at router driving layer. At router driving layer, we utilize the Netfilter framework to intercept UDP packets, which is generated when the two parties use the instant message software to make a voice or video call. The system utilizes the UDP packets as carriers to mix the forged UDP.

- The covert channel has a large capacity. We take the intercepted voice UDP packet as a meta-model to construct a new UDP packet. The forged UDP is constructed by homomorphism using intercepted UDP as a meta-model. It keeps the header consistent and embeds secret information in the payload of forged UDP, which is much larger than information embedded in the specific field (about 2 bits) of the header of the network protocol.
- The covert channel has high concealment: the homomorphic construction allows forged UDP to be checked and modified by firewalls and NAT devices as well as instant message software dedicated servers. And there is no significant traffic change because covert Channel is only created when the usual UDP traffic is heavy.
- We have conducted numerous simulations of covert communication, and successfully transmitted the information transparently from a network-restricted area to a more relaxed area of network supervision, verifying the concealment of the method by installing soft routing as a security gateway on a multi-NIC computer.

2 Related Work

In recent years, relevant researchers have conducted a series of studies in the field of covert communications. Covert channels are characteristically categorized into covert timing channels (CTCs) [4] and covert storage channels (CSCs) [5].

CTC can transfer information to a receiver by modulating the timing behavior of an entity, such as the inter-packet delays (IPDs), or reordering packets of a packet stream. In this respect, the most common entity is the IPD generated by application [6]. Luo et al. [7] designed an IPD-based CTC which maintains normal burstiness patterns of TCP to ensure undetectability. To improve the robustness of the CTC, Wu et al. [8] proposed a CTC where the covert data was compressed by Huffman coding. Based on these, Zhang et al. [9] proposed a covert channel which modulates covert data by the postponing or extending silence periods in VoLTE traffic. Moreover, there are now ways to detect hidden channels in these locations. For example, Aiello et al. [10] proposed a profiling system for DNS tunnels which can be used to detect the covert channel based on DNS.

The researches on CSC mainly focused on the header of the network protocol and the payload of the network packet. Zander et al. [11] utilized TTL of IP header to construct a covert channel. They operate the TTL field of the continuous packet to embed the secret data. Dakhane et al. [12] used the IP ID reference model in the Linux kernel 3.0 to achieve covert communication. They took the Linux kernel and applications to generate new traffic into these packets (TCP), but this is a conceptual model without implementation.

What's more, there are many covert channels based on VoIP. Mazurczyk et al. [13] utilized the header field and data portion of VoIP packet to transmit authentication messages. They embedded the control commands into un-used header fields, and the data portion is embedded in the voice data. However, the embedded control commands are easily detected. Later, they provided a LACK method (Lost Audio Packets Steganography) [14], which provides a hybrid storage-timing covert channel by

utilizing delayed audio packets. LACK uses the multicast of the RTP proto-col and sends the secret data to all the next nodes. LACK is suitable for LAN and has the risk of losing the secret information [15]. Hamdaqa et al. [16] applied a two-phase approach on the LACK steganography mechanism to improve reliability. Furthermore, Nair et al. [17] chose the payload length of the UDP packet as the carrier and realize a packet length channel in chatting applications. Liang et al. [18] proposed a covet channel scheme that communicates by rearranging the packet sending sequences. Their scheme focuses on building packet rearranging covert channel whose function is regardless of the variation on legitimate traffic. Schmidt et al. [19] exploited Voice Activity Detection, which suspends the transmission during the speech to reduce bandwidth requirements. Though they reduced and balanced the bandwidth during the speech, it is difficult to distinguish the noise and silence. These methods usually rely on speech coder and decoder and sometimes there are some errors in parsing and receiving secret data. Generally speaking, G.711 speech coding is more suitable to realize covert communication for VoIP.

In summary, none of them exploits the driving layer of the router and UDP packets generated by the instant message software's voice calling to implement covert communication.

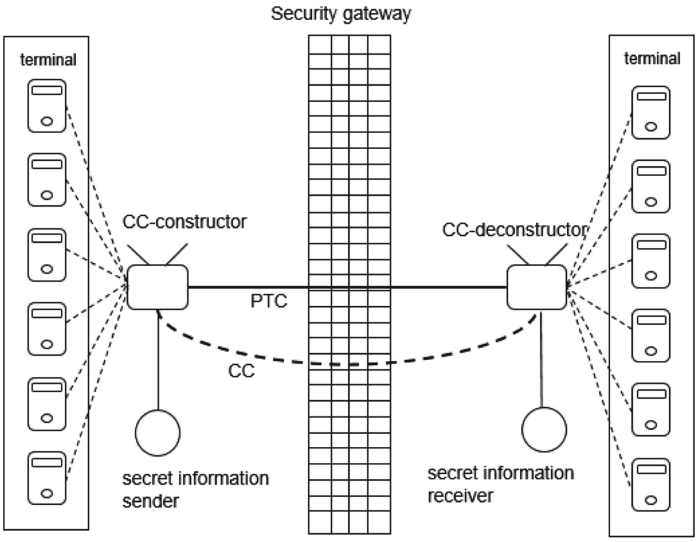

Fig. 2. Framework of covert communication

3 Covert Communication Modeling

The proposed system model for implementing covert communication at the router driver layer is shown in Fig. 2. The terms involved are explained as follows:

- PCC: public carrier channel, can transmit instant messaging voice data and can be used to hide secret information.
- CC: covert channel, is used to conceal information from one end to the other.
- CC-constructor: convert channel constructor, is the sending router that is used for homomorphism construction of covert information. Homomorphism construction refers to constructing a new UDP with the intercepted common UDP as a meta-model and embedding the secret data into the allowable position of the payload.
- CC-deconstructor: covert channel deconstructor, is the receiving router that is used to parse the received secret information.
- Security gateway: The security gateway sets a barrier between the internal network and the external network to prevent internal information leakage. It can protect the network by monitoring, limiting, and modifying the data traffic across the security gateway, shielding the internal information, structure, and operation of the network as much as possible. Specifically, the security gateway can analyze the packet, check the source and destination IP address, source and destination port, service type (such as HTTP or FTP), TTL value, domain name, etc., and monitor traffic changes and suspicious connections. Consequently, the security gateway protects data from being modified, inserted, deleted or replaced by unauthorized parties.

In this model. CC-constructor and CC-deconstructor mount our driver and user application, and the network communication devices can connect the wifi dispatched from the router. Mobile devices such as mobile phones and computers connect to CC-constructor and CC-deconstructor via wifi. The communication devices at both ends will generate a large number of UDP packets when making video or voice calls. At this time, PCC will be established. The secret information sender uploads the secret information to the CC-constructor. When the CC-constructor determines that the UDP traffic reaches a certain threshold, the secret information is split into homomorphic UDP, sent to the CC-deconstructor, and then the CC is established. The establishment of the CC enables the secret information to successfully bypass the check and transmit to the external network without attracting the attention of the security gateway.

4 Implementation

OpenWRT is a GNU/Linux distribution of the highly scalable embedded device. OpenWRT is more than a static firmware image, it is also a complete framework for building custom firmware images which include the boot loader, kernel, root file system, and applications. So its function is very complete and its operating system is easy to modify. Developers can customize an embedded system flexibly that only includes the features they need. In addition, OpenWRT system is more stable and more secure than others. Consequently, we take OpenWRT as the experimental platform. The method of implementing covert communication through OpenWRT mainly includes the following key points: UDP filtering, UDP construction and transmission, and packet information extraction, as shown in Fig. 3.

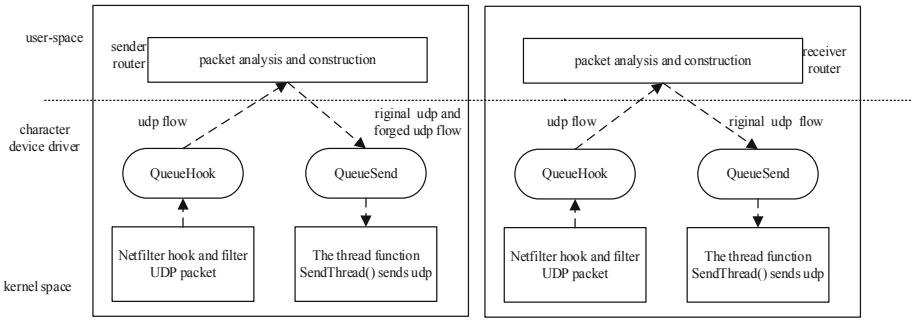

Fig. 3. Framework of covert Communication

In order to improve the stability of the system, an asynchrony should be achieved for drivers and applications. We used a time-stamp to prevent anomaly when the user application crashed unexpectedly. To prevent data interaction conflicts, we designed two circular queues to prevent the conflicts of hooked packets. If the queue is full, the last element will be discarded.

4.1 Protocol analysis

In this paper, we use the instant message software to make a secret information transfer when making a voice call. QQ and WeChat can use both TCP and UDP, but UDP is preferred. When the UDP protocol packet cannot be forwarded normally, they will select the TCP protocol.

After capturing the packets through the Wireshark soft-ware, it can be found that a large amount of UDP packets were generated during WeChat voice and video communications. Analysis of these UDP packets reveals that most of the information in the headers of both parties of the WeChat communication is identical to each other, which includes the source and destination physical addresses, the source and destination network addresses, and the source and destination port numbers, are the same. Therefore, we can use this identification information to construct a new packet that can be used to store secret information at the payload (payload refers to data part) field of the forged packet. The CRC check is performed on the secret information embedded in the payload, and the receiver can use the CRC check code to ensure the correctness of the data. Since the header information of forged packets is the same as that of the packets generated by WeChat communication, it helps the forged UDP pass the security gateway check and no connection anomalies emerge. And only when the UDP traffic passing through the sender router is very large, a small amount of constructed UDP traffic will be mixed in, so there will be no significant change in bandwidth. Moreover, forged UDP packets are not encrypted, so there is no clutter in the form. Even if the attackers intercept packets with secret information, they may also be regarded as ordinary WeChat communication packets. Consequently, the secret transmission is achieved.

In the analysis of UDP packets during the WeChat voice and video communication, we found that the first 20 bytes of the payload portion of these UDP packets have similar characteristics. So when constructing UDP packets, the first 20 bytes of the forged packet payload can be obtained by duplicating the original UDP packet. In order to facilitate the embedding and parsing of a large number of secret data, we fixed the length of the packet and filled the insufficient part with random numbers. And the CRC check code is added after the secret information for identifying the forged packets, as shown in Table 1.

Table 1. Forged packet forma

20 Bytes	The length of valid data	Valid data	Random number	CRC check code

4.2 UDP Filtering

Since WeChat voice and video communication mainly use the UDP protocol, it is necessary to filter out the UDP packets. To implement the filtering of UDP, we write a driver program. In the program, we call a hook function. Its main functions are as follows: the checkpoint packet is stored in Linux sk_buff structure; the head pointer of the data link layer is firstly obtained through the structure, judge whether the network layer protocol of the packet is IP protocol; if not, release the packet; if it is, move the pointer to point to the IP protocol header and judge whether the transport layer protocol of the packet is UDP protocol; if not, the package is released; if it is, the packet will be stored in a circular queue and wait for further processing.

Packets entering the router from outside all will go through the NF_IP_PRE_ROUTING point; all packets sent from the router, including the packets forwarded by the router, will pass through the NF_IP_POST_ROUTING point before entering the network. Therefore, we can register the hook function for these two points that we can intercept all UDP packets generated during WeChat and QQ voice or video communication. In this paper, we register this hook function to the NF_IP_POST_ROUTING point. The reasons for selecting this point will be explained in the next chapter. By mounting the driver program on the OpenWRT router, we can achieve the operation of UDP filtering.

4.3 UDP Construction and Transmission

Because the router core has the low computing power and small storage capacity, the packet needs to be extracted from the kernel's address space to the user's address space for further analysis. In this paper, we adopt the device driver interface.

We extract the packets in the circular queue to user-space for further analysis. If it is found that there are multiple packets with the same physical address, network address, the port number of source and destination, or other the same key information, these packets can be considered to be generated by the voice or video communication of the instant message software. Using these key information, we can construct UDP packets that are very similar to that generated by the instant message software, and add secret

information to constructed new UDP packets. These forged packets are mixed into common UDP data streams and sent back to the kernel. And finally, the kernel will reconstruct these UDP packets into the form of sk_buff structure and send them out to the destination.

Since NF_IP_POST_ROUTING point has a destination address translation function [20], we register a hook function here to filter all packets through this point. We analyze the intercepted packet and find that the source IP, destination IP, source port, and destination port have been changed while the physical addresses of the source and destination have not been changed. Therefore, we just need to do further processing of the physical addresses. Thus, the hook point NF_IP_POST_ROUTING is suitable for filtering UDP packets, which will result in less translation for us.

We write a new function for constructing packets in user-space. Taking the intercepted UDP packets as the meta-model, this function fills the corresponding ETH header, IP protocol header, and UDP protocol header into the same offset position of the forged packets and inserts the secret information in the payload. We check the secret information in the payload by CRC and then add the check code to the payload. For the ETH layer's physical addresses (source and destination), we modify them in kernel space according to the source and destination network addresses passed from user-space. The modification method is as follows:

- We search the ARP table according to the destination network address. If the destination is found, it is clear that the packet is sent to the local area network (LAN). The packet's destination physical address is changed to the physical address corresponding to the destination network address in the ARP table. The source physical address is changed to the same as that of LAN device. In the experimental environment, the network port of LAN network device is br-lan.
- If the destination network address of the packet is not found in ARP table, the packet will send to the wide area network (WAN). Then we look for the gateways IP in routing table; then we use the IP to search in the ARP table, and finally get the physical address of the gateway. We modify the destination physical address as the physical address of the gateway, and modify the source physical address as the physical address of the network interface of the WAN network device. In the experiment, the network port of the WAN network device is eth0.2.

In addition, the checksum also needs to be recalculated. In the driver, the skb_checksum, ip_fast_csum, and csum_tcpudp_magic functions are called to calculate the checksum. After the packet is constructed, it is sent by the dev_queue_xmit function. For intercepted UDP packets, the source and destination physical address of the ETH layer is also replaced as described above and then released.

4.4 Packet Information Extraction

As same as packet filtering, we need to filter out UDP packets through the router. We used the same driver program and mount it on the receiver's router. After the UDP packet is intercepted, it is extracted from the kernel space to user-space for analyzing.

In user-space, we perform CRC check on the data in the payload and then extract the packet with the successful verification, and merge and restore the packets to

complete the secret information extraction. Packets that failed the verification are sent back to the kernel, which achieves the purpose of not affecting the normal UDP traffic transmission. As described in Sect. 4.3 the header of the constructed packet and the specific header of the payload field are consistent with the normal UDP traffic, so they are not detected and recognized by the security gateway. Forged packets are not encrypted and in a normal form, so they won't attract the attention of an attacker. Even if an attacker modifies or deletes our packet, the receiving router will release the modified UDP when there is an error verification of the CRC check.

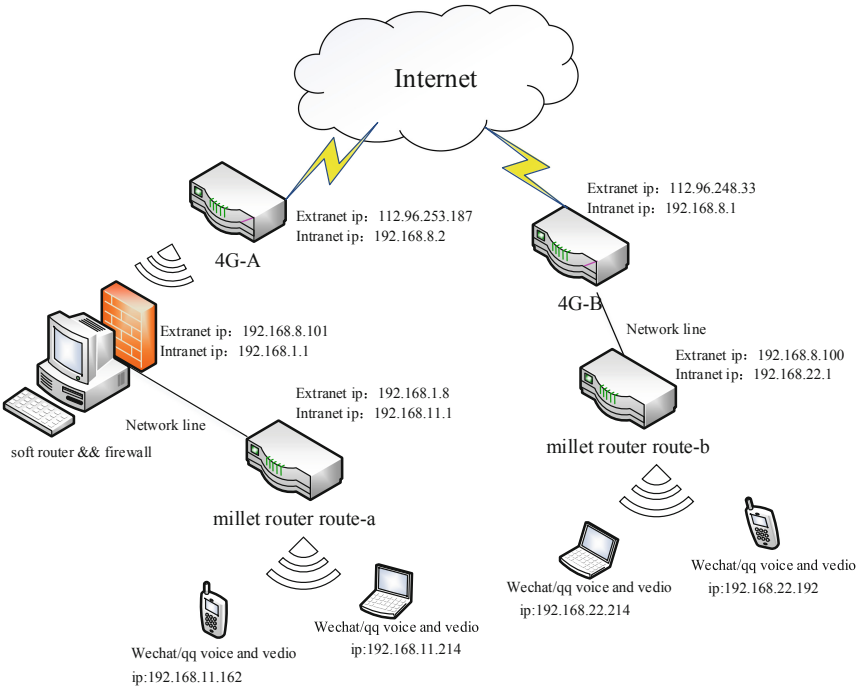

Fig. 4. Experimental environment

5 Experiment Result

In order to verify the feasibility and the concealment of the covert communication proposed in this paper, we conducted the following experiments. The experiments used CC-constructor and CC-deconstructor that were installed OpenWRT system. In the experiment, we applied the character device driver to exchange information between the kernel and user-space. We took WeChat and QQ as the instant message software to make a voice call for generating a large amount of UDP packets. We simulated the real work environment by using the method shown in Fig. 4: 4G-A and 4G-B are ordinary 4G routers that can connect to the Internet. Route-a and route-b are routers with

OpenWRT system which installed driver and user app that we have written. Two mobile phones are used to make video or voice calls in the wifi environment of CC-constructor and CC-deconstructor to simulate real communication. The construction of the security gateway: in the experiment, we performed the following settings to simulate the security gateway. The 64bit Windows 7, which installed with the grass soft router, was used for the security gateway. The grass soft router can perform an online audit, flow control, and some communication behavior control. We set the grass soft router to prevent the interception of outgoing UDP data from port 8000 to port 9000 with IP addresses of 192.168.1.100-192.168.1.200. When the PCC was not established, the TCP/UDP debugging tool failed to send any data whose IP address and port number were within the range above. However, when the PCC was established, the data was successfully sent to the external network.

5.1 Packet Construction and Transmission

In user application, we split the complete secret information into pieces of data that can be sent. Replace the header of the forged packet (ETH header, IP header, UDP header, the first 20 bytes of payload) with the intercepted common UDP header. And then fill the secret data piece and CRC check code in the payload. The forged and com-mon packets are sent back to the kernel and will be reconstructed and sent to the destination. At the other end of the router, we also mounted the driver and intercepted the packet and forwarded it to the TCP & UDP test tool. We took the message "Uh-oh, I am a secret message-_-" for the example to show that how the secret message mixed into the usual message "Hello, this is an ordinary message as the carrier.". The usual messages were sent by the TCP & UDP test tool and the secret messages were generated by the user application. As shown in Fig. 5, all messages intercepted by the receiving router were transmitted to the TCP & UDP test tool.

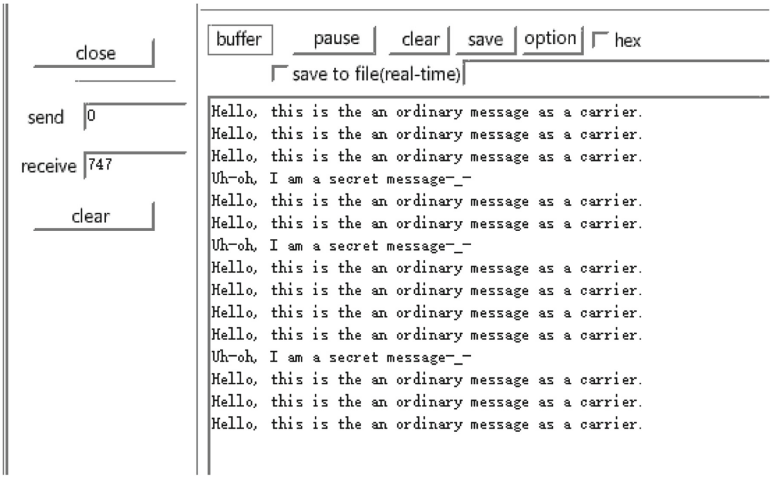

Fig. 5. Forged packets with specific flags

5.2 Packet Information Extraction

In user-space, we embedded information with the content "Uh-oh, I am a secret message-_-" to simulate the secret information. At the other party of the router, we also intercepted all UDP packets passing through the router, and uploaded the packet to user-space. In the user-space, we analyzed the payload field of the packet sent from the kernel and forwarded the valid data of the packet with the successful verification of CRC check to the TCP & UDP test tool. The result is shown in Fig. 6.

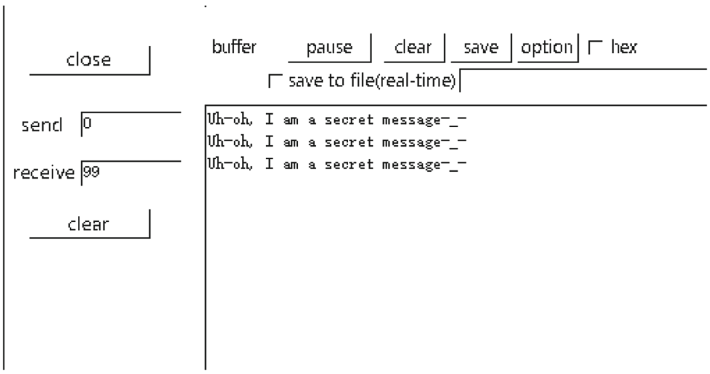

Fig. 6. Intercept forged packets

ETH HEADER	
DEST_MAC:	64:09:80:06:9e:06
SOURCE_MAC:	f4:83:cd:a7:73:30
IP HEADER	
version&hil:45	tos:0
tot_len:00a8	id:0
frag_off:4000	ttl:34
protocol:11	checksum:9905
src ip:7060fdbb	dest ip:c0a816c0
UDP HEADER	
source port:1f40	dest port:c872
PAYLOAD	
5b 00 8c 00 00 00 00 00	61 bb c4 28 00 00 00 00 00
00 55 68 2d 6f 68 2c 20 49	20 61 6d 20 61 20 73 65 63 72
65 74 20 6d 65 73 73 61 67	65 2d 5f 2d 00 25 64 2e 25 64
2e 25 64 2e 25 64 00 6c 6f	63 6b 20 65 e4 aa 8d 21 80 4c
94 35 b7 f9 d3 78 f7 d0 8c	d3 f0 c7 48 38 53 24 5e bb bc
ad 0a 1e d5 dc 1e b9 87 ab	da 07 f8 6e 3c af 68 10 27 5f e0
b3 33 d1 7a 7b 09 cd 9f 67	88 5c 14 93 7a e9 6f 99 a3 f6 44
7d a4 e8 fd 2d	

Fig. 7. Forged packets with specific flags

What's more, we showed the complete packet format including ETH header, IP header, UDP header and the pay-load on the router screen which was put in Fig. 7. The first rectangular box circled the valid data part (from 0x55 to 0x2d), and the second rectangular box circled the CRC check code (0xa4 0xe8 0xfd 0x2d).

5.3 Covert Communication Verification

In order to verify the concealment of the CC, we have con-figured a dual-NIC computer with a Windows7 system as the operating system and installed the grass soft router on it. The grass soft router is a firewall with routing function. Through the configuration, the computer can have the function of the router and firewall.

We first closed all apps on the mobile phone that connected to the router at the side of the grass soft router. At this time, check the number of connections and the connection information on the grass soft router. Then we turned on the WeChat or QQ on the mobile phone to make a voice call. The connection information and UDP traffic are shown in Fig. 8(a) to (d). In the connection interface of the grass soft router, there were some connections and UDP traffic when voice is generated. Then we mounted the driver and application on the OpenWRT routers at both ends. If we dial the WeChat or QQ voice again, we could find that the connection number and connection information are the same as the situation that we did not install the driver and the application. Moreover, UDP traffic change of these connections is not obvious. In Fig. 8(b), the connection between 192.168.1.8 and 157.255.132.53 showed that only 5 Kbps increased than that with no driver. As for the traffic of 233 Kbps, 5 Kbps is negligible. In Fig. 8(d), though the total traffic is smaller that WeChat, the traffic change can also be negligible. That is to say, the firewall does not detect new connections or significant traffic change when sending forged packets. This verifies that the communication is of high concealment.

seq	type	connection type	source IP	source port	dest IP	dest port	total traffic
1	other data	UDP	192.168.1.8	40753	157.255.132.53	16285	233Kbps(145Kbps/87Kbps)
2	QQ	UDP	192.168.1.8	4027	220.249.244.174	8000	200bps(0bps/200bps)
3	DNS	UDP	192.168.1.8	61158	180.76.76.76	53	0bps(0bps/0bps)
4	DNS	UDP	192.168.1.8	54824	180.76.76.76	53	0bps(0bps/0bps)
5	DNS	UDP	192.168.1.8	34911	180.76.76.76	53	0bps(0bps/0bps)
6	not determined	UDP	192.168.1.8	43896	193.228.143.23	123	0bps(0bps/0bps)

(a) dial WeChat voice

seq	type	connection type	source IP	source port	dest IP	dest port	total traffic
1	other data	UDP	192.168.1.8	40753	157.255.132.53	16285	238Kbps(148Kbps/82Kbps)
2	DNS	UDP	192.168.1.8	34911	180.76.76.76	53	0bps(0bps/0bps)
3	DNS	UDP	192.168.1.8	61158	180.76.76.76	53	0bps(0bps/0bps)
4	not determined	UDP	192.168.1.8	43896	193.228.143.23	123	0bps(0bps/0bps)
5	QQ	UDP	192.168.1.8	4027	220.249.244.174	8000	200bps(0bps/200bps)
6	DNS	UDP	192.168.1.8	54824	180.76.76.76	53	0bps(0bps/0bps)

(b) dial WeChat voice and send forgd UDP

seq	type	connection type	source IP	source port	dest IP	dest port	total traffic
1	QQ	UDP	192.168.1.8	53692	163.177.93.149	8000	122Kbps(50Kbps/71Kbps)
2	other data	UDP	192.168.1.8	48653	59.175.19.185	56191	2Kbps(944bps/1Kbps)
3	QQ	UDP	192.168.1.8	4027	220.249.244.174	8000	600bps(0bps/600bps)
4	DNS	UDP	192.168.1.8	27139	192.168.8.2	53	0bps(0bps/0bps)
5	DNS	UDP	192.168.1.8	34556	192.168.8.2	53	0bps(0bps/0bps)
6	DNS	UDP	192.168.1.8	14041	192.168.8.2	53	0bps(0bps/0bps)
7	DNS	UDP	192.168.1.8	57522	111.206.63.16	53	0bps(0bps/0bps)
8	DNS	UDP	192.168.1.8	55398	192.168.8.2	53	0bps(0bps/0bps)
9	DNS	UDP	192.168.1.8	58578	111.206.63.16	53	0bps(0bps/0bps)
10	DNS	UDP	192.168.1.8	55941	111.206.63.16	53	0bps(0bps/0bps)
11	DNS	UDP	192.168.1.8	55942	111.206.63.16	53	0bps(0bps/0bps)
12	DNS	UDP	192.168.1.8	56362	192.168.8.2	53	0bps(0bps/0bps)
13	DNS	UDP	192.168.1.8	14834	192.168.8.2	53	0bps(0bps/0bps)
14	DNS	UDP	192.168.1.8	64598	111.206.63.16	53	0bps(0bps/0bps)
15	DNS	UDP	192.168.1.8	65137	111.206.63.16	53	0bps(0bps/0bps)

(c) dial QQ voice

seq	type	connection type	source IP	source port	dest IP	dest port	total traffic
1	QQ	UDP	192.168.1.8	53692	163.177.93.149	8000	126Kbps(51Kbps/74Kbps)
2	other data	UDP	192.168.1.8	48653	59.175.19.185	56191	1Kbps(200bps/834Kbps)
3	QQ	UDP	192.168.1.8	4027	220.249.244.174	8000	534bps(0bps/534bps)
4	DNS	UDP	192.168.1.8	55941	111.206.63.16	53	0bps(0bps/0bps)
5	DNS	UDP	192.168.1.8	34556	192.168.8.2	53	0bps(0bps/0bps)
6	DNS	UDP	192.168.1.8	14041	192.168.8.2	53	0bps(0bps/0bps)
7	DNS	UDP	192.168.1.8	57522	111.206.63.16	53	0bps(0bps/0bps)
8	DNS	UDP	192.168.1.8	55398	192.168.8.2	53	0bps(0bps/0bps)
9	DNS	UDP	192.168.1.8	64598	111.206.63.16	53	0bps(0bps/0bps)
10	DNS	UDP	192.168.1.8	58578	111.206.63.16	53	0bps(0bps/0bps)
11	DNS	UDP	192.168.1.8	55942	111.206.63.16	53	0bps(0bps/0bps)
12	DNS	UDP	192.168.1.8	65137	111.206.63.16	53	0bps(0bps/0bps)
13	DNS	UDP	192.168.1.8	14834	192.168.8.2	53	0bps(0bps/0bps)
14	DNS	UDP	192.168.1.8	56362	192.168.8.2	53	0bps(0bps/0bps)
15	DNS	UDP	192.168.1.8	27139	192.168.8.2	53	0bps(0bps/0bps)

(d) dial QQ voice and send forged UDP

Fig. 8. Detected network connection

5.4 Performance

In the experiment, we used 1.09 MB information as test data to test the transmission efficiency. The constructed UDP contains 100Bytes of private data.

As shown in Fig. 9, it can be seen that with the sending speed becomes larger within 20 UDP/s to 50 UDP/s, the time for sending complete hidden information is gradually shortened. However, when the speed exceeds the 50UDP/s, the time required increases with the increase of the packet trans-mission speed. The anomaly is due to the fact that with the increase of the sending speed, the UDP loss is aggravated, and the time of resending UDP increase. The dotted curve in the figure indicates the number of packets received by the receiver per second. Therefore, the speed of receiving UDP decreases. In the entire system, the system efficiency is the highest when the UDP transmission speed is 50 UDP/s (5 KB/s). The number of UDP sent per second exceeding 50 will cause the communication to become blocked, and the UDP loss rate will increase, making CC-constructor and CC-deconstructor worse.

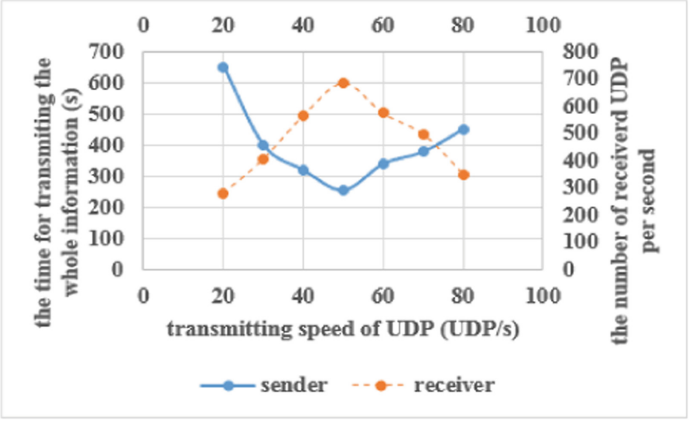

Fig. 9. The comparison of different information hiding

We also investigated other information hiding methods and compared them with ours. The results are shown in Table 2. It shows that information hiding in the field of network protocol' header is of high real-time but with a small capacity. The methods of JPEG and BMP are of low real-time and the capacity is not large too. By contrast, our method based on the router driver is of large-capacity and higher real-time.

Table 2. The performance of the existing covert communications

Information hiding methods	Carrier	Capacity	Real-time
The hiding in JPEG image	JPEG image	One byte per 2.5 bytes	Low
The hiding in audio	Audio	16 bps (phase coding)	High
The hiding based on LSB algorithm	24-bit BMP image	One byte per 8 usual bytes	Low
The hiding in IP header	The fields such as checksum, identification, TTL, TOS, etc.	About 2 bytes per TCP packet	High
The hiding in TCP header	The fields such as ISN, ACK, RST, etc.	One byte per TCP packet	High
The hiding in ICMP header	The field of time-stamp and ID	One byte per ICMP packet	High
Our method	The payload of a new constructed UDP	100 bytes per UDP packet (40 Kbps)	High

6 Conclusion

Based on the CC' large-capacity, we have first embedded the secret information in the UDP data stream generated by the voice or video call of the instant message software at CC-constructor. Furthermore, we add CRC check code in the payload of forged UDP at CC-constructor, and check the CRC code at CC-deconstructor for the extraction. The feasibility and concealment of the method have been perfectly proved through experiments. Furthermore, we have built firewall between CC-constructor and CC-deconstructor. No new connections or obvious data flow changes are detected, which can prove our method of high concealment. The downside of our system is that we have only studied two kinds of instant message tools WeChat and QQ to verify our system so the universality of our system still needs to be improved.

In a further study, we will analyze UDP data on more instant message software such as Skype, MSN, UC, and others to develop a self-adaptive covert communication system. In addition, our method is likely to be applicable to other Linux-based platforms. Therefore, we plan to investigate the possibility of applying our method to other router platforms to find its universalities, such as SuperWRT or RouterOS.

References

1. Lampson, B.W.: A note on the confinement problem. Commun. ACM **16**(10), 613–615 (1973)
2. Chuanchuan, G.: Research and implementation on the system of storage network covert channels detection, Ph.D. dissertation, University of Electronic Science and Technology (2016)

3. Mazurczyk, W., Szczypiorski, K.: Steganography of VoIP streams. In: Meersman, R., Tari, Z. (eds.) OTM 2008. LNCS, vol. 5332, pp. 1001–1018. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88873-4_6
4. Luo, X., Chan, E.W.W., Chang, R.K.C.: TCP covert timing channels: design and detection. In: 2008 IEEE International Conference on Dependable Systems and Networks with FTCS and DCC (DSN), pp. 420–429. IEEE (2008)
5. Chow, J., Li, X., Mountrouidou, X.: Raising flags: detecting covert storage channels using relative entropy. In: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 25–30. IEEE (2017)
6. Zhang, X., Tan, Y.A., Liang, C., et al.: A covert channel over volte via adjusting silence periods. IEEE Access **6**, 9292–9302 (2018)
7. Peng, T., Liu, Q., Meng, D., et al.: Collaborative trajectory privacy preserving scheme in location-based services. Inf. Sci. **387**, 165–179 (2017)
8. Xiao, Y., Chen, H.H., Du, X., et al.: Stream-based cipher feedback mode in wireless error channel. IEEE Trans. Wirel. Commun. **8**(2), 622–626 (2009)
9. Zhang, X., Tan, Y., Xue, Y., et al.: Cryptographic key protection against FROST for mobile devices. Cluster Comput. **20**(3), 2393–2402 (2017)
10. Aiello, M., Mongelli, M., Cambiaso, E., et al.: Profiling DNS tunneling attacks with PCA and mutual information. Log. J. IGPL **24**(6), 957–970 (2016)
11. Zander, S., Armitage, G., Branch, P.: Covert channels in the IP time to live field (2006)
12. Zander, S., Armitage, G., Branch, P.: Dynamics of the IP time to live field in Internet traffic flows (2007)
13. Alsaffar, H., Johnson, D.: Covert channel using the IP timestamp option of an IPv4 packet. In: Proceedings of the International Conference on Electrical and Bio-medical Engineering, pp. 48–51 (2015)
14. Guan, X.X., Wang, C.D., Li, Z.G., et al.: Reliable and double-blind IP covert timing channel. J. Comput. Appl. **6**, 40 (2012)
15. Mazurczyk, W., Kotulski, Z.: New security and control protocol for VoIP based on steganography and digital watermarking. arXiv preprint cs/0602042 (2006)
16. Mazurczyk, W.: Lost audio packets steganography: the first practical evaluation. Secur. Commun. Netw. **5**(12), 1394–1403 (2012)
17. Nair, A.S., Kumar, A., Sur, A., et al.: Length based network steganography using UDP protocol. In: 2011 IEEE 3rd International Conference on Communication Software and Networks, pp. 726–730. IEEE (2011)
18. Liang, C., Wang, X., Zhang, X., et al.: A payload-dependent packet rearranging covert channel for mobile VoIP traffic. Inf. Sci. **465**, 162–173 (2018)
19. Schmidt, S.S., Mazurczyk, W., Keller, J., et al.: A new data-hiding approach for IP telephony applications with silence suppression. In: Proceedings of the 12th International Conference on Availability, Reliability and Security, p. 83. ACM (2017)
20. Yun, L.: Packet filtering algorithm based on netfilter under linux. Comput. Eng. **35**(11), 143–145 (2009)