Chapter 9
A RouterUpdate Method for Tor Anonymous Communication System

Tianbo Lu, Bing Xu, Shixian Du, Lingling Zhao and Xiaomeng Zhang

Abstract Among all the anonymous communication systems, Onion Routing is most widely used. Tor affords users with anonymous service in communication. It can be used in running anonymous web browsing and announcement, real-time communications, IRC, SSH and other TCP applications. After analyzing the source code of the Tor system, this paper introduced the network layout, working flow, the RouterUpdate method of establishing a virtual circuit and data sending or receiving in Tor system. The method helps Tor client using the relay nodes which have been stored to connect the internet of Tor without connecting the list server. Then the paper introduced a method to help the Tor client using all the applications whether or not using the SOCKS.

Keywords Anonymous communication · Onion routing · Tor · SOCKS · Directory server

9.1 Introduction

Information vulnerabilities gradually become a security hidden danger to people's life in net. Anonymous communication is an effective method to guard users' privacy, which can protect bilateral identities and communication relations from being obtained by attackers.

Based on MIX [1], an idea of multiple step objective route, transmitting data via many middle nodes is proposed. Onion routing is an efficient method in anonymous communication [2–4]. Tor, a second generation of onion routing, has been widely used [5–10]. Tor is used to keep watch on flow filter and sniff analysis,

T. Lu (✉) · B. Xu · S. Du · L. Zhao · X. Zhang
School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China
e-mail: lutb@bupt.edu.cn

W. E. Wong and T. Ma (eds.), *Emerging Technologies for Information Systems, Computing, and Management*, Lecture Notes in Electrical Engineering 236, DOI: 10.1007/978-1-4614-7010-6_9,

making communication in overlay network composed of onion routers, and to realize anonymous external links, anonymous hide and so on. As Tor is a network of virtual passageways, it makes an anonymous foundation of a series of application. So people can share information in public networks without caring privacy may be violated. As Tor disperses user flow to many different places in internet, there is no single point that links a user and his destination. In this way, Tor helps decline risk of simple and high-level flow analysis.

The paper is organized as follows. Section 9.1 gives an overview of Tor. Section 9.2 presents the overall structure and the primary flow. Section 9.3 mainly shows the design and related algorithms of store routing nodes. Section 9.4 introduces the design and application about rewriting network function.

9.2 Overall Architecture

The procedure to build a virtual circuit is shown in [4]. Based on the introduction, let us make a summary of a primary Tor process.

In an anonymous system, the user who wants to hide his identity should start OP process, which is responsible to build communication links and encrypt and decipher data. It obtains node information from Directory Server and selects one from the set of Tor nodes to consult secret keys, and last it builds a safe information channel with the former node. The building process conforms to short-term Diffe-Hellman secret key exchange protocol, as well as the TLS which guards the privacy of information channels and the security of information retransmission further. Then all data should be transmitted in this channel. Next OP goes on expanding to other Tor nodes via the built channel and exchanging secret key to build a multilayer encrypted channel. Data should be encrypted according to the order of Tor nodes from the later layer to the former one. During the transmission process, every time encrypted data passes a Tor node, it will be deciphered. It won't stop until the data of the last node has been transmitted to the destination. In the process data goes back from destination, it will be encrypted one time when passing a Tor node, deciphered when arriving at OP and transmitted to the application program finally. As each Tor node only has its own encrypting and deciphering keys, external attackers and Tor cooperators won't obtain the plaintext of the communication data only if they could obtain secret key of all nodes in the route.

9.3 RouterUpdate Method Based on Voting Mechanism

As the available onion router list and the neighboring network information are obtained from the directory server, when a directory server cannot be connected for some reason, the Tor OP cannot get node information and it would be difficult

to connect to Tor network, thus losing the ability for anonymous communication. This has been the defects of Tor against certain network firewall.

9.3.1 RouterUpdate Method Design

The RouterUpdate (RU) method is divided into several steps.

- The routerlist initialization and load.

Loading part of this method is after the Tor initialization and before read the network information document on the directory server. After Tor OP sends request information to the directory server, if it is unable to obtain the document, the method will automatically select the network and routerlist in the loading document. After that, Tor OP can create virtual circuit through the routerlist.

- Analyze and collect the ORs in the current Tor network, dynamically selecting the optimized OR and adding into the pre-defined global list.

By checking the current storage router node information, the status of the Tor network relay nodes in the records will be inspected. The Tor relay node with better efficiency and performance will be elected by screening algorithm of RU method. Then, the nodes filtered out will be stored in the temporary routerlist.

- The third step is periodically writing a custom OR global list into the hard disk, and optimizing the store files.

In the main loop of the system, set a timer and counter. When the timer times out, RU will automatically write data to disk and counter will be plus one. When the number of writes in counter records is greater than 3, the counter will be cleared automatically, and store files in the disk will be optimized.

9.3.2 Custom Routerlist Loading

- Loading Point Processing

In the Tor system design, there is a pattern for the load of the routerlist. A typical loading process occurs in the initialization process, as follows:

(1) The system will try to load the network status recently used. (2) After an available network status document is obtained, the Tor client will load the OR indicated in the document into the client's routerlist after some necessary checks, and check the available nodes and download the RC of some OR if needed. (3) After the nodes loaded is the creation of the virtual circuit and the subsequent operation.

- Disk Data Reading

Client will access a file with the similar format of the default store file on disk. The default file name is "cached-best Routers". At this point, the client needs to define a metadata for store files operation to save the corresponding document, and to associate the router lists in files and programs. In this example, the client will define it as:

desc_store_t
fname_base: holds the name of the description file
fname_alt_base: holds the name of the backup file
mmap: point to the file data entry
description: This document textual description
type: the type of document
journal_len: the length of the document log
store_len: the length of the document.

- File Reading

Reading the file occurs during the router list initialization. RU method can read the file when the routing list is not assigned. The client will connect the global list and the predefined store file by setting desc_store_t during the initialization. Note that the initialization of the list and desc_store_t is carried out simultaneously.

9.3.3 Real-time Data Collection

- Store in the main loop

The Tor client will periodically maintain some everyday events. Every time unit, Tor will run some detection and maintenance work. Collection of router state is exactly suitable for periodic operation, so the client will add this feature to run_scheduled_events with the daily maintenance function. This program will be called once at intervals of 1 s.

- Collection standards

On the basis of the optimized node located, the client will maintain a smartlist with MAX_NUM_BESTNODES size to record the ORs which have the max bandwidth in all ORs, dynamically stores in the temporary router list. Bandwidth here is the average of known bandwidth and maximum acceptable bandwidth.

9.3.4 Writing into Disk

Each time after the collection and analysis of data, and every standard time passed, it will write standards-compliant nodes to disk.

9.3.5 Optimizing File

Because the client has been writing router information, the store file is bound to storage repeated even invalid router information. When loaded it will result in the router list bloated and inefficient. System optimizes the router information every three time to write, and every time system load the file, and automatically optimizes the routing information.

9.4 Expand Tor SCOKS Agency

Some application programs which do not support SOCKS can't use the anonymous service afforded by Tor and other services. The method of "Extend for Tor" (EFT) in this paper aims to allow those application programs to access the network by Tor as agency, but won't alter those programs.

9.4.1 The Design of EFT Method

EFT method can modify dynamically linked libraries by setting the LD_PRELOAD environment variable, so that make it point to custom library of EFT method. If resetting default dynamically linked libraries at exit, this can run Tor, at the same time, and can let it automatically load in each progress space of executable program.

In the library file, this module rewrite normal connect function and select function, as the Fig. 9.1 shows.

- About the process of establishing connection

EFT method make an application automatically call connect function which is defined by module when building TCP connection, so as to submit right of control to Tor OP, rather than rock-bottom protocol. Later, application can firstly establish connection with Tor OP, and then establish connection with destination address.

- About the process of data transmission

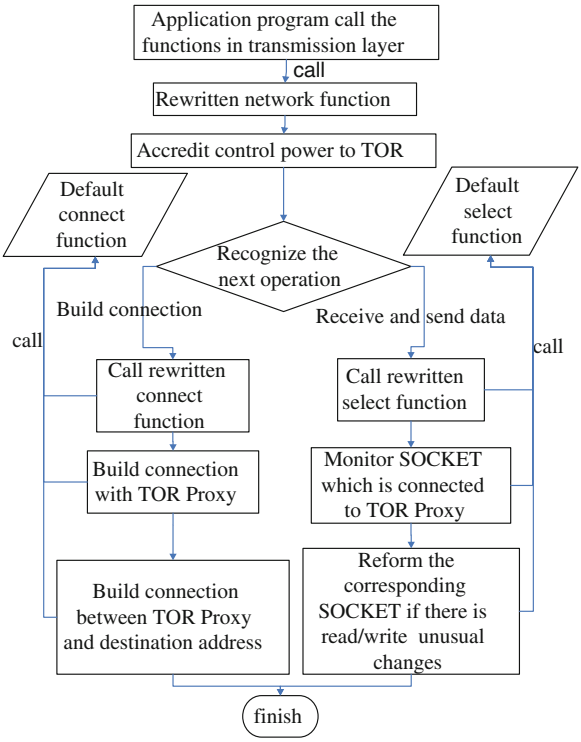

Fig. 9.1 Application call network function

This method can make rewritten select function inform Tor OP at first and submit the right of control to Tor when the state of data flow which is concerned with Tor system, instead of previous SOCKS. So the data flow is forced to go through Tor OP.

9.4.2 The Implementation Process of EFT Method

- Set range of application of Tor OP

Tor client need to be able to distinguish local network from external network. User can declare local address gateway and external address gateway by himself. By default, local address is localhost, others are external address. In addition, User can also declare corresponding address to use Tor OP by himself. And we can declare it in the file of PROXYconfig (Fig. 9.2).

- Data sending and receiving

The basic strategy of data sending is shown in the above figure. Tor OP plays a role in connecting the sender and receiver.

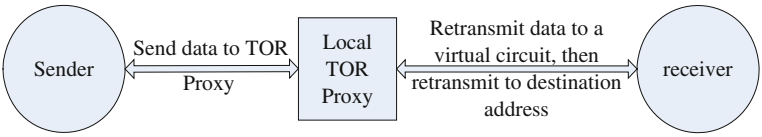

Fig. 9.2 The transmission process through Tor OP

Under the request of receiving data, if client may use the disposal way of no blocking, in turn, the module should rewrite select function.

- About select function

It will achieve the monitoring of related SOCKET. When SOCKET need to read and write, module will know it by its defined select function and submit the message to the corresponding function to deal with. Notice that data firstly send to the Tor OP, then according to the destination address, Tor OP send it through virtual circuit.

9.5 Conclusion

Based on the study of source code of Tor system, the article analyzes anonymous communication system of Tor and its principle and introduces the design and application about rewriting network function. As it is difficult for Tor users to do further work in the condition that it could not be connected to Directory Server, this paper suggests collecting network data in User and sponsoring the connection project autonomously. However, there are still some points to be improved, such as information collection of User, collection algorithm and security of Tor system.

Acknowledgments This work is supported by the following programs: the National Natural Science Foundation of China under Grant No. 61170273; Research Innovation Program for Young People of China Beijing University of Posts and Telecommunications with title “Code Security Assurance”; 2010 Information Security Program of China National Development and Reform Commission with the title “Testing Usability and Security of Network Service Software”.

References

1. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
2. Goldschlag, D.M., Reed, M.G.. Syverson, P.F.: Hiding routing information. In: Proceedings of Information Hiding, vol. 1174, pp. 137–150 (1996)
3. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Onion routing for anonymous and private internet connections. Commun. ACM. **42**(2), 39–41 (1999)

4. Reed, M.G., Syverson, P.F., Goldschlag, D.M.: Anonymous connections and onion routing. IEEE J. Sel. Areas Commun. **16**(4), 482–494 (1998)
5. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: Proceedings of USENIX Security Symposium, pp. 21–21 (2004)
6. Edman, M., Syverson, P.F.: AS-awareness in Tor path selection. In: Proceedings of the 2009 ACM Conference on Computer and Communications Security, pp. 380–389 (2009)
7. Evans, N., Dingledine, R., Grothoff, C.: A practical congestion attack on tor using long paths. In: Proceedings of the 18th USENIX Security Symposium (2009)
8. McLachlan, J., Tran, A., Hopper, N., Kim, Y.: Scalable onion routing with Torsk. In: Proceedings of CCS, pp. 590–599 (2009)
9. Jansen, R., Hopper, N., Kim, Y.: Recruiting new tor relays with BRAIDS. In: Proceedings of the 2010 ACM Conference on Computer and Communications Security, pp. 319–328 (2010)
10. Tang, C., Goldberg, I.: An improved algorithm for Tor circuit scheduling. In: Proceedings of the 2010 ACM Conference on Computer and Communications Security, pp. 329–339 (2010)