International Journal of Future Generation Communication and Networking
Vol. 9, No. 8 (2016), pp. 345-362
http://dx.doi.org/10.14257/ijfgcn.2016.9.8.33

Code Analysis and Improvement of Onion Routing Anonymous Systems

Tianbo Lu, Youwen Wang, Lingling Zhao, Yang Lin and Xiaoyan Zhang

School of Software Engineering, Beijing University of Posts and Telecommunications, 100876, Beijing, China
lutb@bupt.edu.cn, wang6087480@126.com, wodepengyouzhao@163.com

Abstract

With the development of Internet technology, network based activities such as e-commerce, internet voting and e-government, etc. have become increasingly frequent, People are increasingly concerned about the identity of network activity, content and other private information. In order to protect user privacy in network communications, governments, companies, universities and research institutes are pushing the research and development of the onion routing systems (TOR). Tor is the most popular anonymous communication system currently, which is based on technology of the second-generation onion routing. Tor has a low latency, encrypted data transmission, secure channel, etc., which are widely used in anonymous Web browsing, instant messaging, Secure Shell Client (SSH), etc. However, the development of the onion routing systems is constrained by complicated code factors. According to this situation, this paper put forward an overall architecture of TOR code, which contains Application Needs, Code Analysis and Improvement. This paper summarizes the overall framework of tor code structure, in order to make the code structure clearer, the whole code module is divided into several sub modules, using the function call diagram and UML diagram illustrates the main function of each module and call relation between each module. This paper gives code analysis for anonymous architecture of TOR. Finally, this paper put forward the improvement of routing algorithms of TOR. Based on this paper, a method has been created which is used to understand the tor code easily. Furthermore, systematic analysis of the tor code provided in this paper aims to research the working principle and promote the further development of the onion routing.

Keywords: *The Onion Routing (TOR), Anonymous Theories, Code Analysis, Routing Algorithm*

1. Introduction

In recent years, the development of computer technology and network technology has brought great convenience to people's life. With the continuous improvement of computer data handling capacity and the rapid development of data communications capability, along with the increase of the demand for all kinds of communication software, people also pay more attention to the security and anonymity of the communication software, there is a growing emphasis on network security and privacy protection. The existing network security technology is basically only concerned with the data itself, while ignored identity information of the communication entity, which leading to an attacker, through traffic analysis or other eavesdrop means, can obtain identity information between the two communication sides easily, this creates a great threat to the privacy of the user's personal identity. It is such a demand, the anonymous communication system came into being, which main purpose is to hide the identity of the communicating parties or hide communication relationship, under these conditions, the onion routing system arises at the historic moment. Tor [1] is a network of virtual tunnels that allows people and groups to

ISSN: 2233-7857 IJFGCN

improve their privacy and security on the Internet, and it is also an open source software, thus attracted government, companies, universities and research institutions pushing the research and development of the onion routing systems.

However, because the source not only has a huge code structure, but also has a complex relationship of calling, and there is no very detailed code analysis material, these factors have greatly limited the understanding of the source code, which affect the further development of TOR software. The purpose of this paper is to analyze the code structure of the tor, the whole tor code module is decomposed into several sub-modules, through function calls diagram and UML diagram to analyze the main function of each module and call relation between each module. Routing algorithms for tor code at the same time also to do the corresponding improvement, no longer according to the bandwidth choice routing nodes, this paper proposes a routing algorithm of random and uniform distribution, thus increase the cost of the attacker steal information, and to some extent, improved the tor network anonymity and security.

2. The Entry of the Program

Most of the source program entrance is the main function. In order to achieve unit tests more simply, the main function of tor systems will call tor main function, for the latter is the main function of the Tor system .these relations can be seen in tor_main function. The file describes the relationship between the main and tor_main with very few lines. The main function is the top level module. Handles signals, multiplexes between connections, implements main loop, and drives scheduled events. Figure 1 is the call graph of the main function. There are two important functions in the main function, which is the do_main_loop function and tor_init function.

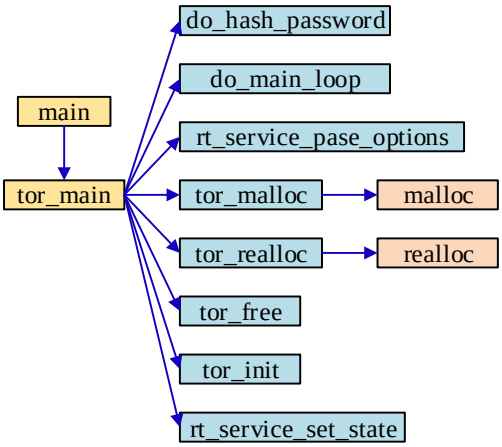

Figure 1. Call Graph of the Main Function

2.1. The do_main_loop function

The main loop included some important initialization parts, such as DNS processing, signal processing, keys, cell pool, token bucket, network configuration and the multithreading configuration. For the client of tor system, some parts have no corresponding processing, actually directly to discuss the client's initialization section below. Figure 2 is the call graph of the do_main_loop function.

1) Initialization of signal processing

letter processing mechanism, sock sensing mechanism and callback mechanism is completed by the ibrary of libevent, handle_signals function will register each signal and corresponding signal processing functions and parameters in the libevent system, signal

processing functions of main process is signal_callback function, which values is a signal parameter. Signal processing of child process of tor is realized by signal action function.

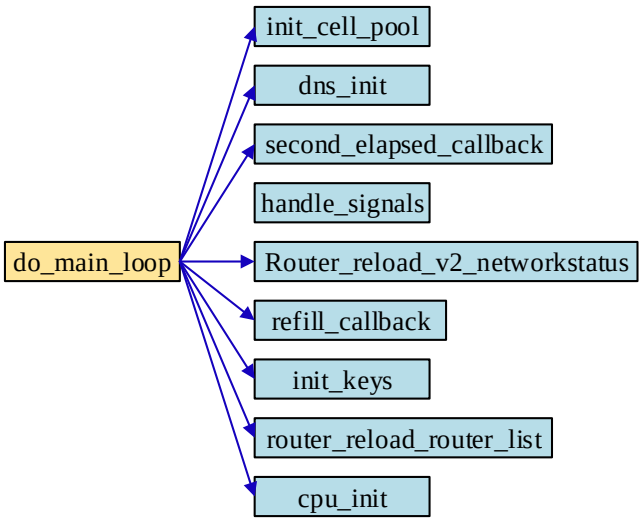

Figure 2. Call Graph of the do_main_loop Function

2) Initialization of key and TLS

Tor system uses three keys, encryption, authenticaiton and coordination, the encryption is the symmetric key between OR and OR, or symmetric Key between OP and OR, the uthenticaiton is the public Key, which is used to deal with exchange protocol of DH Key between OP and OR [2]. The coordination belongs to Identity key of the all. The directory Server have extra directory signature Key, the client has a separate client identity key.

For the client, the client identity Key can be generated temporarily, can be discarded after use it. So, the initialization process is to generate the client identity key simply, then initialize the context of TLS [3]. For the Server, the Server Identity Key is the Key of long-term preservation, need to read from key file saved, if starting sever for the first time, generation key and save it (save it in the datadirectory/keys generally). In general, the process of the key initialization and the TLS initialization, is initialization to all the identity of the tor system, done for the normal operation of key distribution of tor system in cryptography. The code as below:

```
01. if (!server_mode(options)) {
02.   if (!(prkey = crypto_pk_new()))
03.     return -1;
04.   if (crypto_pk_generate_key(prkey)) {
05.     crypto_pk_free(prkey);
06.     return -1;
07.   }
08.   set_client_identity_key(prkey);
09.   /* Create a TLS context. */
10.   if (router_initialize_tls_context() < 0) {
11.     log_err(LD_GENERAL,"Error creating TLS context for Tor client.");
12.     return -1;
13.   }
14.   return 0;
15. }
```

Finally, after the execution of the above code, system will set up two global variables of client_identitykey and client_tls_context.

3) Initialization of cell pool

Cell is basic data unit of protocol communication, the specific organization of the unit has been described in detail of form in design file of tor system [4]. In a recent version of the tor system, the specific format and related command of cell also has changed a little, but there is no big change in the basic ideas and format. In order to control cell unified, the cell pool is designed. The initialization is an initialization of global cell pool, and set the global variable cell_pool finally.

4) The core of the main loop

The core of the main loop mainly done a couple of things, the second callback settings, the callback setting of the token bucket refill; the main loop of event set. Second callback Settings is prepared to safeguard the Tor system. As the name suggests, the second correction is that the tor system will perform callback function every second. Its core is implemented by the timing event of the libevent library. Token bucket callback will play a role when buffer event function has not been activated, which is activated by default. Which will read the configuration parameter to set a fixed time, then perform the callback function. Which core is also implemented by the timing event of the libevent. The content of Refill_callback function is an increasing operation of token number. The code as below:

```
01.  for (;;) {
02.    //All active linked conns should get their read events activated.
03.    SMARTLIST_FOREACH(active_linked_connection_lst, connection_t ,
04.                      conn,event_active(conn->read_event, EV_READ, 1));
05.    called_loop_once = smartlist_len(active_linked_connection_lst) ? 1 : 0;
06.    update_approx_time(time(NULL));
07.    // poll until we have an event, or the second ends,
08.    // or until we have some active linked connections to trigger events for.
09.    loop_result = event_base_loop(tor_libevent_get_base(),
10.                      called_loop_once ? EVLOOP_ONCE : 0);
11.  }
```

When the system performs the do_main_loop function, remove all events from the active linked list, then added to the event pool for the libevent scheduling. If there are connections in active linked list, the scheduling of libevent event pool only perform once and return.

2.2 The tor_init Function

The tor_init function is the main initialization function of the tor system, which have several important global variables, these variables include time_of_process_start, connection_array, closeable_connection_lst and active_linked_connection_lst. time_of_process_start is on behalf of the really run time to start of process, connection_array is all connections list, including the DIR, AP, OR, EXIT and other types of connections, closeable_connection_lst is the connection list that system want to close, active_linked_connection_lst is all of the active linked list, including DIR, AP and other types of connections.

This function initializes the three child function, the first one is the rep_hist_init function, which initialize the reputation history, including the mapping table, bandwidth array and forecast port initialization. The second one is rend_cache_init function, which initialize the service descriptor cache. The last one is address_map_init function, which initializes the address mapping. In addition to this, there is a configuration file that can

initialization tor systems, which can generate global_options and options_init_from_torrc of global options variable. Figure 3 is the call graph of the tor_init function. Through this call graph, it can be very vivid description of the relationship between each function call.

3. The Establishment of the Link

After initialization of the tor system, then the communication link start to establish. Before analyzing the code, first introduced the following entities:

(1) User client (User): It is computer that anonymous communication with the outside world through the onion routing network, which usually connected to the Tor network through an onion User proxy.

(2) Onion Proxy (OP): It is an agent service programs that run in the local system, which is responsible for establishing the transmission channel of data, and sending the client's data to the tor network or from accessing data from the server of the tor network.

(3) Onion Router (OR): It is the onion routing server, also can be called the intermediate node, which main task is to establish anonymous communication link and forward data. At the time of creating anonymous link, each onion router establish a connection with next onion router through the Transport Layer Security (TLS). In addition, is revoked, changes and extensions of the link are also completed by the onion router. Onion routing not only forward data, but also participate in the work of encryption and decryption of data. Therefore, it is the core part of the tor system.

(4) Router Directory Server: Mainly be responsible for the collection, management and storing of OR information of all the nodes.

(5) Bridge Directory Server: When the client can't get direct access to anonymous service of the tor network, can request it by mail or web page for access to the bridge nodes task of encryption and decryption of data. Therefore, it is the core part of the tor system [5].

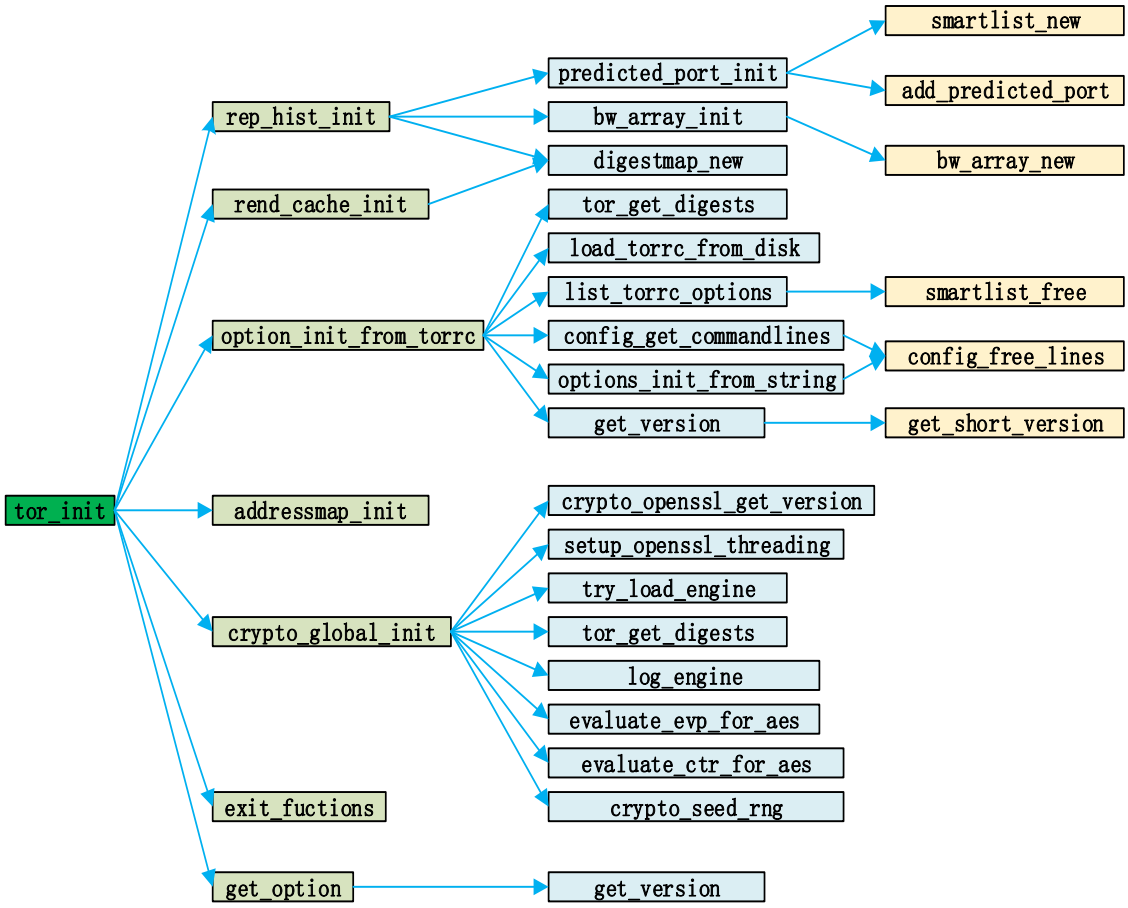

Figure 3 Call Graph of the tor_init Function

(6) Entry Node: It is the first onion routers on the path, which provides direct communication with the user.

(7) Exit Node: It is the end onion routers on the path, which provides direct communication with the target server.

(8) Middle Node: It is the onion routers on the path that in addition to the entry and exit node.

In the stage of establishing a connection, the OP will download status information of network from the directory server through HTTP protocol, then Establishing an anonymous communication link based on the transport of TCP [6]. Tor network structure diagram as shown in Figure 4.

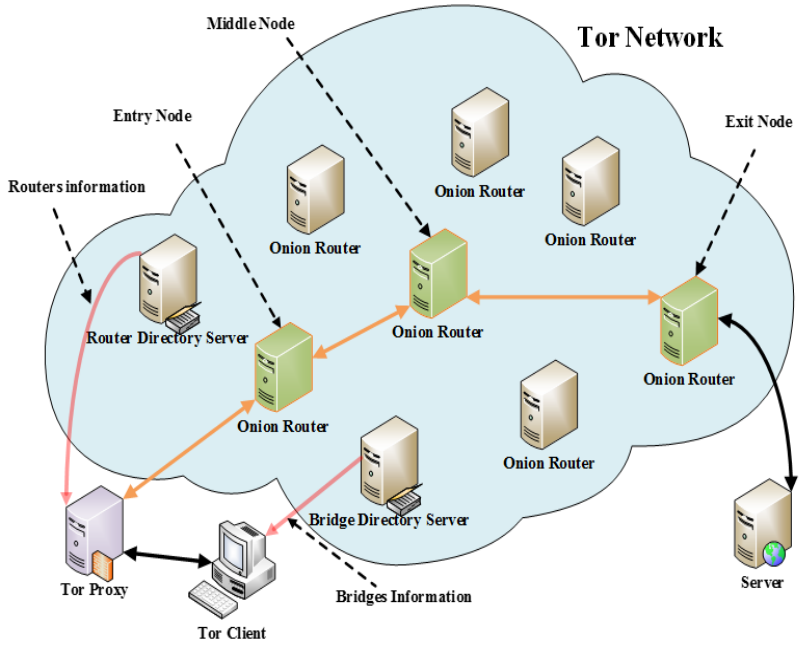

Figure 4. The Graph of tor Network Structure

When the OP have downloaded the routing information from the directory server, then choosing a OR to join the communication line according to the predetermined algorithm, at the same time, OP and OR use Diffie-Hellman handshake negotiate a key, which used for generating symmetric encryption to establish communication links and encrypted communication data, and in order to prevent the attacker to tamper with data, tor agent and three tor routing establish a connection with the TLS (Trans Port Layers security) connection [7]. In the process of establishing the link, the user's OP will incremental build lines with a progress of a hop each time. The code as below:

```
01. origin_circuit_t *
02. circuit_establish_circuit(uint8_t purpose, extend_info_t *exit, int flags)
03. {
04.   circ = origin_circuit_init(purpose, flags); //Initializes the link structure
05.
06.   if (onion_pick_cpath_exit(circ, exit) < 0 || //Choose exit node
07.       onion_populate_cpath(circ) < 0) { //Choose other nodes  of link
08.   }
09.   //build the first link node of the link nodes
10.   if ((err_reason = circuit_handle_first_hop(circ)) < 0) {
11.   }
12.   return circ;
13. }
```

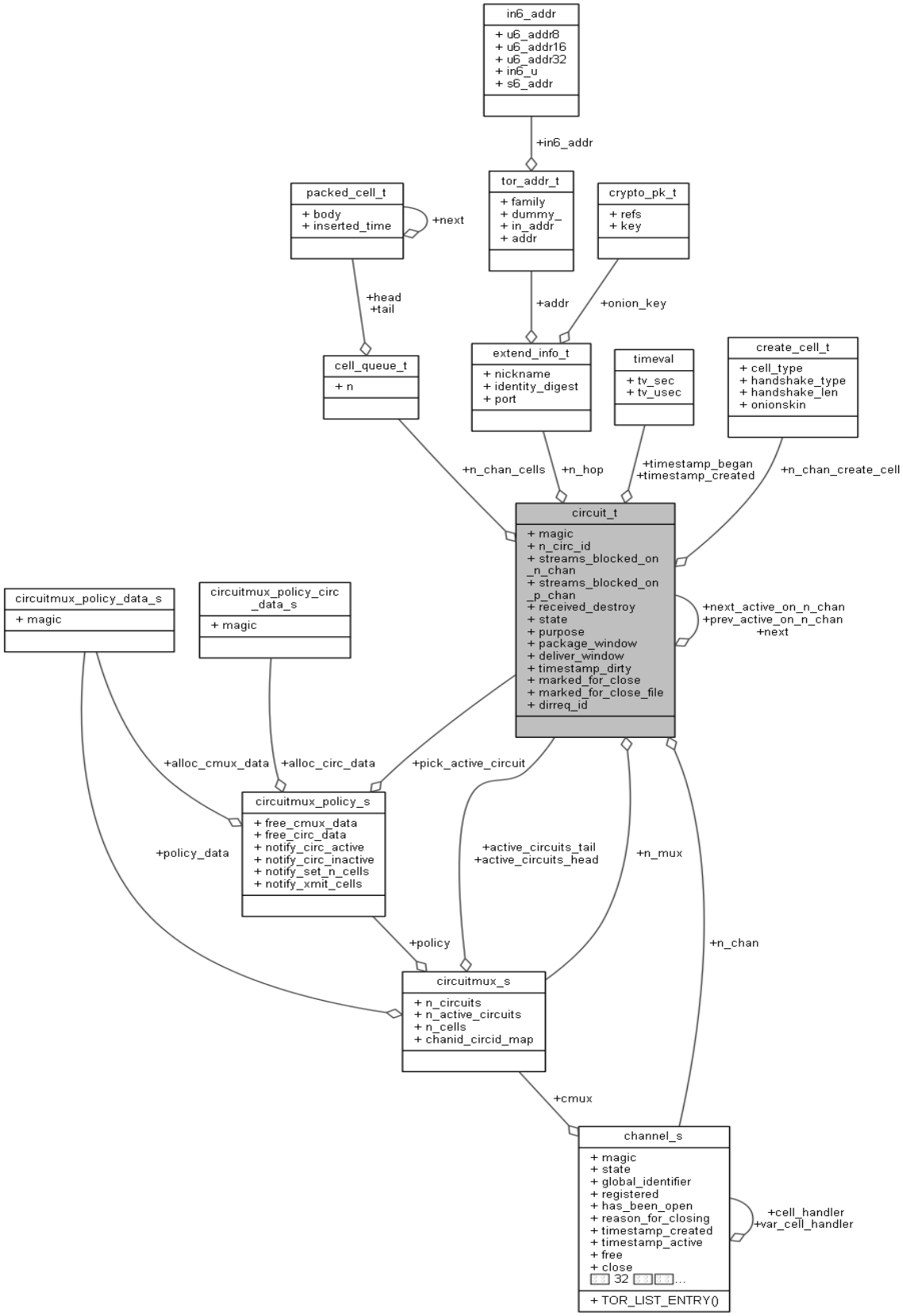

Figure 5 The UML Diagram of circuit_t Structure

A circuit is a path over the onion routing network. Applications can connect to one end of the circuit, and can create exit connections at the other end of the circuit. AP and exit connections have only one circuit associated with them (these connection types are closed when the circuit is closed), whereas OR connections multiplex many circuits at once, and stay standing even when there are no circuits running over them [8].

There's a very important structure in this function, it's a circuit_t structure, which can fill one of two roles. First, an or_circuit_t links two connections together: Either an edge connection and an OR connection, or two OR connections. (When joined to an OR connection, a circuit_t affects only that cells sent to a particular circID on that connection. When joined to an edge connection, a circuit_t affects all data.)Second, an origin_circuit_t holds the cipher keys and state for sending data along a given circuit [9]. At the OP, it has a sequence of ciphers, each of which is shared with a single OR along the circuit. Separate ciphers are used for data going "forward" (away from the OP) and "backward" (towards the OP). At the OR, a circuit has only two stream ciphers: one for data going forward, and one for data going backward. The UML diagram of circuit_t structure as shown in Figure 5.

The establishment of the anonymous communication link is the first step of initialization running environment of tor, mainly used to determine the identity of the anonymous forwarding nodes and onion key of encrypting data. First, the OP choose an OR as an entrance node of path, then random select other OR to extend the link, OR to extend the link has been to expand the link to the last jump OR nodes, until the link to the last jump of OR nodes, the last node as export node, which will be directly connected to the target server, Such a safe anonymous encryption channel based on the reroute is established [10]. The sequential figure of Tor's circuit building process as shown in Figure 6. In the establishment process of new virtual circuit, user's OP will build lines incrementally at each hop schedule.

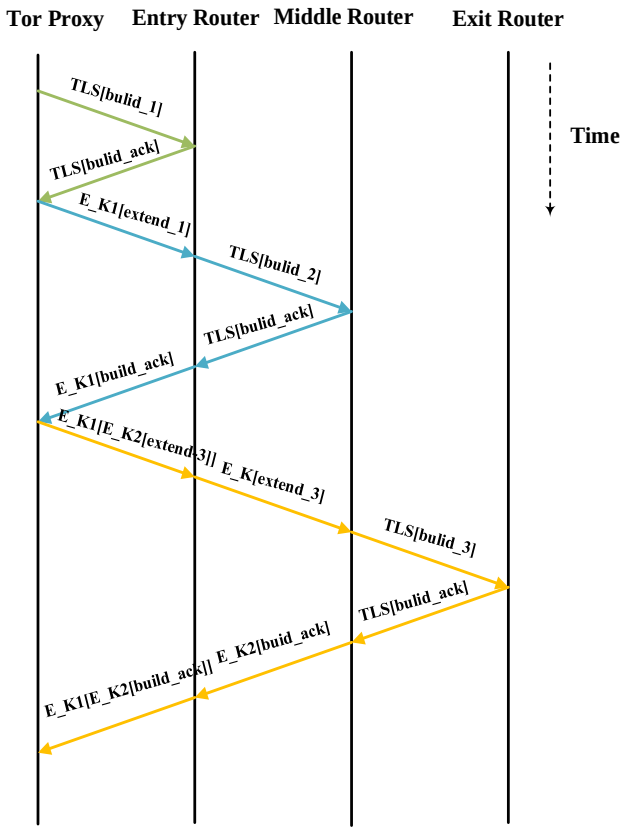

Figure 6. Circuit Building Process

In order to balance the anonymity and cost, OP choose three routing nodes constitute a circuit. The OP and three routing nodes are connected by TLS (Transport Layers security), which is to prevent an attacker from tampering with the data as well as to prevent an attacker simulating the route selected. Meanwhile, in order to prevent the middle attack (man-in- middle attack). Tor proxy and each routing node of the circuit uses the Diffie-Hellman handshake to negotiate a key, which is used for symmetric encryption (DES, AES, *etc*.), when the client and server make a communication, client data is encrypted three times using three key negotiated, it means that three layer is equivalent to the data. When the data reaches the entrance route, which will decrypt the date with key1, it means the first layer of data is stripped, then successively stripped two and triple layers in the middle route and exit route, so data became the unencrypted raw data at the exit route, the export route can record these unencrypted data, the problem is one of the biggest problems in tor. In the establishment process of circuit, first, the client chooses the first hop along the circuit [11]. Then the chosen entry router forwarding the client's request to the chosen middle router. Last the entry and middle routers forwarding the final circuit building request message to the desired exit node. Key K1 is a shared secret key between the client and the entry router. Key K2 is a shared secret key between the client and the middle router.

4. The Transmission of Encrypted Data

The packets of transmission of tor network is called the Onion bag (Onion), which use agent technology to realize the communication with the target host, through the use of this kind of agent technology, can make the Onion bag independently applications with a variety of different network protocols [12]. All packets of HTTP, SMTP and FTP can turned into a unified data structure through onion agent, it is convenient to transmission in the Tor network [13]. So, when the user using Tor to complete an application, without making any changes, onion agent technology can help users to process, this make it easy to use for the user. The code as below:

```
01. int
02. onion_skin_TAP_create(crypto_pk_t *dest_router_key,
03.                       crypto_dh_t **handshake_state_out,
04.                       char *onion_skin_out) /* TAP_ONIONSKIN_CHALLENGE_LEN bytes */
05. {
06.   char challenge[DH_KEY_LEN];
07.   crypto_dh_t *dh = NULL;
08.   int dhbytes, pkbytes;
09.   tor_assert(dest_router_key);
10.   tor_assert(handshake_state_out);
11.   tor_assert(onion_skin_out);
12.   *handshake_state_out = NULL;
13.   memset(onion_skin_out, 0, TAP_ONIONSKIN_CHALLENGE_LEN);
14.   dhbytes = crypto_dh_get_bytes(dh);
15.   pkbytes = (int) crypto_pk_keysize(dest_router_key);
16.   tor_assert(dhbytes == 128);
17.   tor_assert(pkbytes == 128);
18.   note_crypto_pk_op(ENC_ONIONSKIN);
19.   /* set meeting point, meeting cookie, etc here. Leave zero for now. */
20.   memwipe(challenge, 0, sizeof(challenge));
21.   *handshake_state_out = dh;
22.   return 0;
23. }
```

In the phase of data transmission, the OP send encrypted packet to the next-hop of OR nodes. Every node on the path are randomly selected by the user. In the following Figure 7,

we selected an anonymous communication link, which consists of OP and three OR nodes, displaying a transmission process of data. As can be seen from the figure, the sender is the first node in the whole link, the recipient is the last node the whole article link in the message sender is the first node. However, the anonymous system does not contain the recipient, anonymous path is over at the exit router note [14]. The Entry Routing Node is the next-hop node of OP, which is also an entry node of link, the Exit Router Node is the last-hop node of link, which is also an exit node of link.

As to reverse transfer of response data, which is the recipient sends the response data from recipient to sender, is the reverse process followed the above procedure. The recipient sends the data to exit router, then encrypt the data unit, encryption algorithm and key are different with that used by the above transmitted procedure [15]. Finally, the OP of client Decrypt the data multiply and Plaintext restructuring, then send the data to the sender.

Among the whole circuit, intermediate nodes only know the next-hop node and previous-hop node of itself, on hop and OR nodes, enabling a secret identity of communicants. Tor will packet data units into the same length, which Consisted of a header and a payload, if less than one data unit, then filled packet to the same length using bag-filling technique. This is done to prevent an attacker capture analysis, Because of inconsistent length of data and lead to attract attacker special attention of data. When the data flow that formed with the fixed-size data units through each intermediate node of circuit, each data unit will be decrypt by OR using the public key of symmetric. Therefore, decryption consumes most of the time at the data transmission process, which became the main reason for the delay of tor system [16].

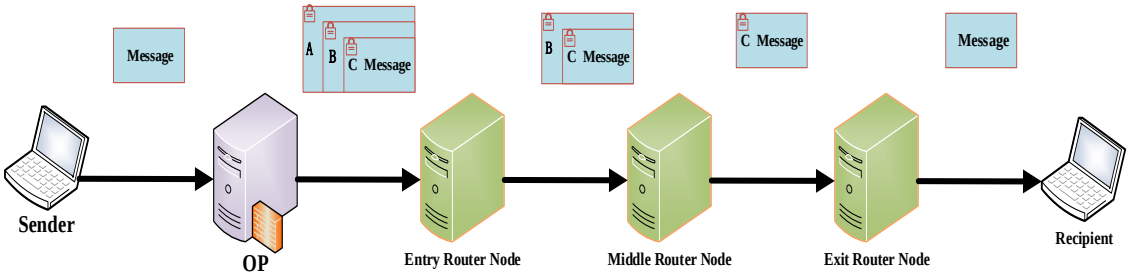

Figure 7. The Diagram of Transmission Process of Message

For the reverse transmission of response data, that is the response data from the receiver to the sender, which is contrary to the above process, the recipient sends the data to Exit Router Node, which break it down into small data unit, and then the data unit is encrypted by the intermediate router nodes, the encryption algorithm and key are different with that of the forward transmission [17]. Finally, the OP decrypted the message multiply and reorganize plaintext, and send the data to the sender. Among the whole link, OR node only know the previous-hop and the next-hop OR nodes, in order to achieve the secret identity of communicants. Tor will packet data units into the same length. Generally speaking, each data unit has 512Bytes, which is constituted by a header and a payload. If the data unit less than 512 Bytes, then bag-filling technique is used to fill the packet to the same length, and thus can lead to the attacker capture and analysis the packet.

5. The Routing Algorithms of Tor

At present, the tor routing algorithm consists of three parts, the first part is the selection algorithm of the entry node, the second part is the selection algorithm the intermediate node, the third part is the selection algorithm of the exit node. Here, through the analysis of the current Routing Algorithm of the source code, we illustrate the original Routing

Algorithm separately in Routing Algorithm I, Routing Algorithm II, Routing Algorithm III[18]. Figure 8 is the call graph of the routing.

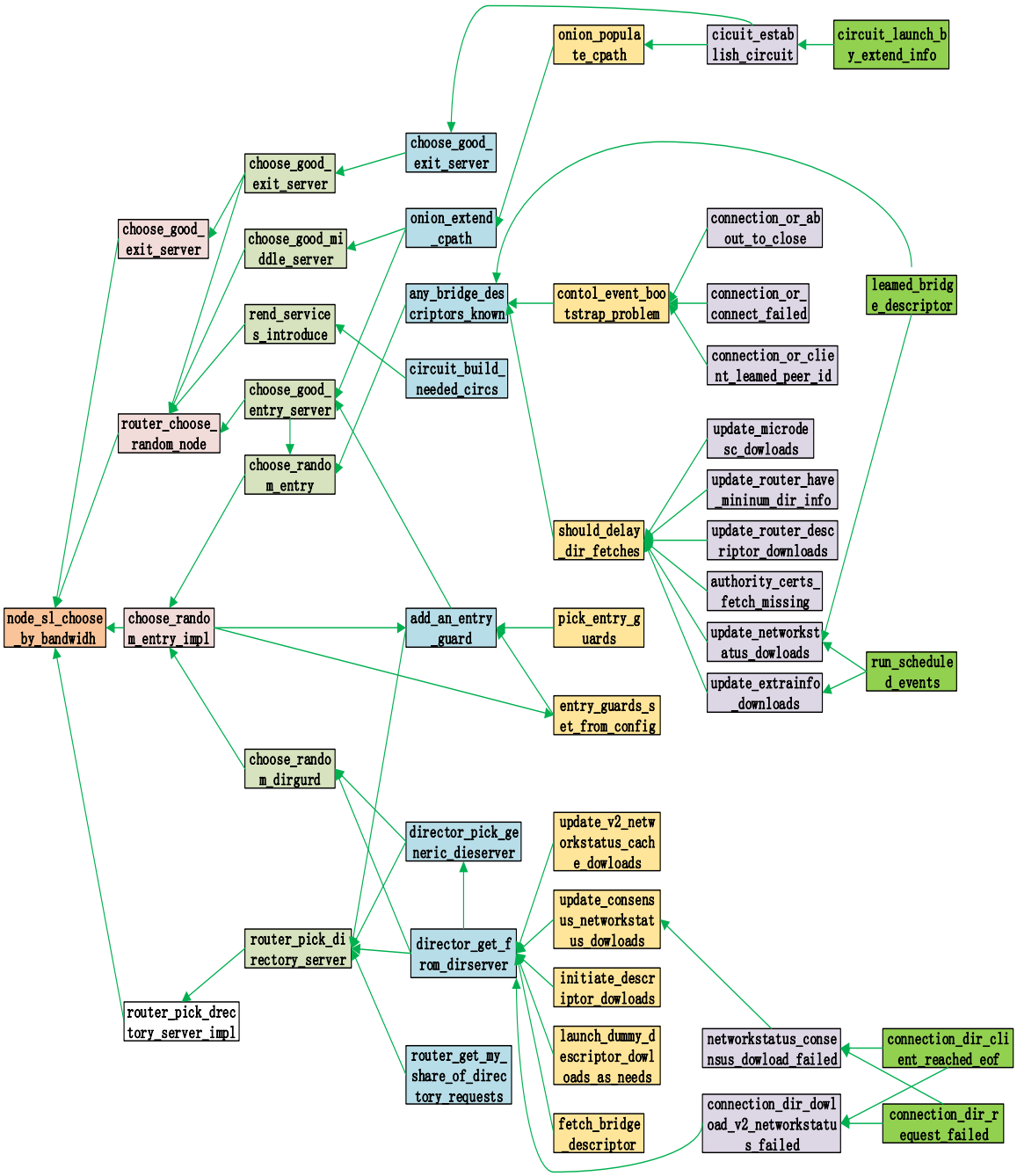

Figure 8. Call Graph of the Routing

5.1. Routing Algorithm I: Selection Entry Node for the Path

In tor system, the first OR nodes are called entry node, the security of the entry node is relating to safety of the whole link and the sender's anonymous, security of entry node is critical, therefore, we must establish a secure entry node. Usually select a group of stable and rapid OR node as an alternative entry node through a directory server. The bandwidth of These OR is bigger than the average bandwidth of all the OR, meanwhile uptime should be above of average uptime. The directory server stored these information of OR on the

hard disk permanently [19]. When a problem occurs at an entry node, the directory server will randomly select a new OR quickly to replace that one, to ensure security, stability and normal operation entry node. Execute steps as the following:

```
01. Export a possible list of entry node;
02. Choose three as a reference of the entry node;
03. Entry Guard choose one node from the three reference node;
```

5.2 Routing Algorithm II: Selection Middle Node for the Path

The choice of middle node is mainly based on bandwidth and uptime. In the Tor network, some OR nodes of the high bandwidth and high stability are often used. According to the bandwidth value of the onion router to select the intermediate nodes [20]. Directory server classified the bandwidth value according to the bandwidth of state information of OR, the bandwidth value is divided into five sections with the unit of number of bytes transmitted per second. In each interval, set a counter $\mathrm{R[i]}, (\mathrm{i} = 1,2,3,4,5)$,which can record the number of nodes in each interval, and a counter $\mathrm{Count[i]}, (\mathrm{i} = 1,2,3,4,5)$, which can record the number of used nodes in each interval, as long as the OR node is used in this interval, the corresponding counter is incremented one count. Besides, each interval set different value of weight, $\mathrm{weight[i]}, (\mathrm{i} = 1,2,3,4,5)$, a larger weighting value is given to an interval of larger bandwidth, in order to increase its probability of being selected. When the directory server choosing intermediate nodes, according to the frequency of use, the number and the weight of OR nodes values to calculate the assessed value of the selected nodes, namely:

$$\mathrm{Value[i]} = \begin{cases} \mathrm{Value[i]} = +\infty & \mathrm{R[i]} = 0 \\ \mathrm{Value[i]} = \dfrac{Count[i]}{R[i] \times Weight[i]} & R[\mathrm{i}] \neq 0 \end{cases} \qquad (\mathrm{i} = 1,2,3,4,5)$$

The plan divided OR nodes into different zones according different values OR bandwidth, and then given different weights values of nodes according the actual number of OR node to be used finally calculated the evaluation value to choose the intermediate nodes. This selection algorithm can choose an OR node of stronger processing ability, higher bandwidth, better performance to establish the anonymous communication link at a higher probability [21]. Execute steps as the following:

```
01. Eliminate entry and exit nodes that has been selected;
02. Eliminate the client (in order to prevent the client itself is a Tor routing nodes);
03. Export series of running node that meet the conditions;
04. If need fast node, then
05.     Use the optimal bandwidth Algorithm that is similar to step 5 of Routing Algorithm III;
06. Else
07.     Random select exit nodes;
```

5.3 Routing Algorithm III: Selection Exit Node for the Path

The exit node in the tor system is the last OR node of the anonymous path, the data is fully decrypted into clear date here, then sent to the receiver. As the data is decrypted completely, so its safety is very important for the Tor network. Therefore, Tor system set up three options of the exit node:

(1) The node selection scheme of open type: the nodes selected as this scheme can connected to any address without any system.

(2) The node selection scheme of middle type: the nodes selected as this scheme can only be limited to forwarding data, and can't connect to the target host.

(3)The node selection scheme of private type: the nodes selected as this scheme only allow you to connect to a given host or network.

Currently, most of the OR choose the selection scheme of middle type and private type, these selection scheme allows exit node connected to the majority of the target host, at the same time, can resist the attacker eavesdropping and traffic analysis, thus can ensure the safety of the exit node [22]. Execute steps as the following:

```
01. Eliminate the current client as a possible exit node;
02. Eliminate Non-Running state node and the node that run time cannot meet the requirements;
03. Eliminate those nodes that don't declared as an exit node;
04. Based on bandwidth measurement, export node list that meet the requirements;
05. Using the weighting algorithm of bandwidth to choose an exit node;
```

By source analysis, we found three routes that they are all in optional collection, through the RAND_byte function of encryption algorithm of OpenSSL, priority selected node randomly based on bandwidth, which is used for load balancing. In addition, tor routing nodes of high-bandwidth can handle more services. The benefits of this kind of random selection is one of the best anonymity and security. The reason of such high anonymity is that routing algorithm has the largest route candidate set, and the reason of such high security is that the routing nodes selected by routing algorithm are usually in different countries, thus the possibility of being captured at the same time is very low, but it will also reduce the performance of the circuit [23]. For example, the entry node in Asia, middle node in Europe, exit node in Africa, which no doubt would lead to high latency.

6. The Improvement of Routing Algorithm

Currently, the node bandwidth recorded by directory server is mainly self-reported in the Tor network, thus gives the attacker an entrance of attacking the link, malicious nodes can report to the directory server false bandwidth that is much larger than their actual bandwidth, which can induce more link through the malicious nodes. Therefore, in the following we will present a random routing algorithms based on uniform distribution of bandwidth to improve the performance of the circuit.

When using this selection algorithm, the probability of a router is chosen is based on its bandwidth range, rather than the bandwidth value. That is to say, the attacker can't simply rely on increasing bandwidth to get more opportunities of being selected, because the selection is random in bandwidth range. If attacker want to obtain a higher chances, can only increase the number of nodes in the network, this greatly increased the cost and difficult of attack. Here, the user can choose a routing algorithms according to their different mission, the choose focus is between anonymity and performance, in order to adapt to the practical application. According to the above problem, put forward a function, which is defined as follows:

$$f_a(x) = x^a, x \in [0,1] \qquad (1-1)$$

Figure 9 is the distribution of function $f_a(x)$ in the interval $[0,1]$, where $\lim_{a\to 1} f_a(x) = x$.We get bandwidth values of relay nodes through the directory server, assuming that each node in the Tor network is safe and reliable, and its stated bandwidth are true and correct. Suppose node sorted from small to large in accordance with the bandwidth and numbered from 0 to N - 1, The index table of node is $[n \times f_a(x)]$ ([x] is maximum integer that not exceeding x), Where x is a random value of [0,1]. The function is controlled by a value of a, which means that we can set a different value to control the index table of routing, when the user wants to select those high-grade routing node, who can select a lower value of a, in order to sacrifice some performance, increase anonymity. When the user wants to select those low-grade routing node, who can select a higher value of a, in order to sacrifice some anonymity, increase performance.

As can be seen, based on this routing algorithm can effectively divide the node bandwidth to grade range, introduces a variable x, making the routing nodes are no longer

concentrated in a few high-bandwidth, but based on the bandwidth range; while introduces a control variables a, making the user can select different anonymity and network performance based on their different applications [24]. The routing algorithm process of using a random uniform distribution is as follows:

(1) Obtain the bandwidth of the relay node from the directory server.
(2) Sort the Bandwidth of relay nodes and numbered from 0 to n-1.
(3) Randomly selected a value x from [0, 1].
(4) Computing $[n \times f(x, l)]$, then selected routing node.
(5) Establishing a communication connection according to the selected routing nodes.

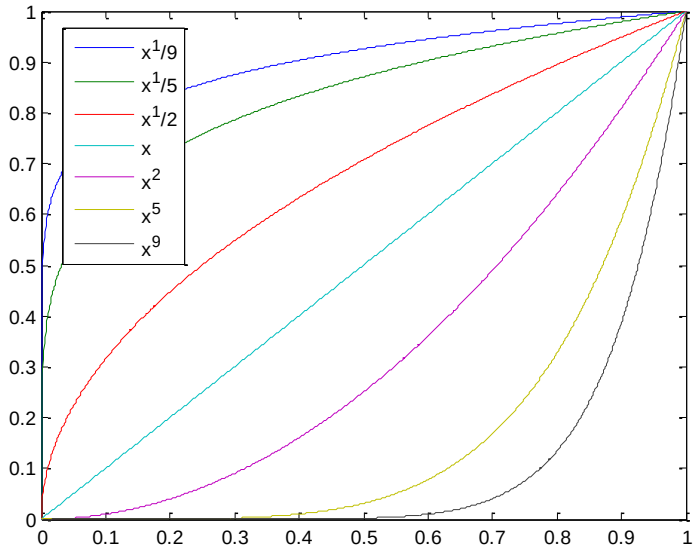

Figure 9. Distributions of Functions by Different Levels

7. Conclusion

Along with the network more and more involved in people's production and life, people also pay more and more attention to network security and privacy. The research purpose of anonymous communication is that it can hidden the communication relation between two sides through a certain method, so an attacker can't associate the communicating parties or direct access to the communication side. Currently, the study of anonymous communication become a hot topic in the field of network security research. This paper studies the Tor anonymous communication systems , which is most widely used at present, we detailed analysis of the Tor code structure using of UML diagrams and call graph of function, focusing on analysis of the initialization process, the main loop of the program, the process of establishing the link and the process of data transmission, *etc*. at the same time puts forward the corresponding improvement for the source routing algorithms, proposes an improved routing algorithm based on random uniform distribution for the weakness of its existing route, allow users to select the balance of anonymity and performance depending on the application providing a choice depending on the different application. Compared to the original routing algorithm of Tor network, to some extent, improved routing algorithm improved the security and anonymity of network.

As a result of our efforts, it is seen that the research of tor is far from mature for the routing algorithm and there are still many challenges facing designers, operators and researchers. This is unsatisfactory, and hopefully, by providing an overview of the literature efforts done, the overview will contribute in providing reference for researcher in the area of TOR.

Acknowledgements

This work is supported by the following programs: the National Natural Science Foundation of China under Grant No.61170273; 2010 Information Security Program of China National Development and Reform Commission with the title "Testing Usability and Security of Network Service Software".

References

[1] R. Dingledine, N. Mathewson and P. Syverson , "Tor: The second-generation onion router", Naval Research Lab Washington DC, **(2004)**.
[2] M. G. Reed, P. F. Syverson and D. M. Goldschlag, "Anonymous connections and onion routing", Selected Areas in Communications, IEEE Journal, vol. 16, no. 4, **(1998)**, pp. 482-494.
[3] D. Goldschlag, M. Reed and P. Syverson, "Onion routing Communications of the ACM", vol. 42, no. 2, **(1999)**, pp. 39-41.
[4] P. Syverson, G. Tsudik and M. Reed, "Towards an analysis of onion routing security", Designing Privacy Enhancing Technologies. Springer Berlin Heidelberg, **(2001)**, pp. 96-114.
[5] P. F. Syverson, D. M. Goldschlag and M. G. Reed, "Anonymous connections and onion routing", Security and Privacy, 1997, Proceedings, 1997 IEEE Symposium on IEEE, **(1997)**, pp. 44-54.
[6] D. McCoy, K. Bauer and D. Grunwald, "Shining light in dark places: Understanding the Tor network", Privacy Enhancing Technologies, Springer Berlin Heidelberg, **(2008)**, pp. 63-76.
[7] R. Snader and N. Borisov, "A Tune-up for Tor: Improving Security and Performance in the Tor Network", NDSS, **(2008)**.
[8] M. G. Reed, P. F. Syverson and D. M. Goldschlag, "Onion routing network for securely moving data through communication networks", U.S. Patent 6,266,704. **(2001)**.
[9] R. Dingledine and N. Mathewson, "TOR http://tor.eff.org/cvs/doc/tor-spec.txt, **(2008)**.
[10] K. Loesing, "Measuring the Tor Network from Public Directory Information", 2nd Hot Topics in Privacy Enhancing Technologies, **(2009)**.
[11] M. Perry, "Tor Flow: Tor network analysis", http://fscked.org/ TorFlow-HotPETS-final.pdf, **(2010)**.
[12] P. F. Syverson, M. G. Reed and D. M. Goldschlag, "Onion Routing access configurations", DARPA Information Survivability Conference and Exposition, 2000, DISCEX'00, Proceedings IEEE, **(2000)**, pp. 34-40.
[13] J. Camenisch and A. Lysyanskaya, "A formal treatment of onion routing", Advances in cryptology–CRYPTO 2005, Springer Berlin Heidelberg, **(2005)**, pp. 169-187.
[14] R. Dingledine, M. Freedman and D. Molnar, "The Free Haven Project", Workshop on Design Issues in Anonymity and Unobservability, **(2000)**.
[15] A. Jones, "Anonymous communication on the internet", Indiana University, **(2004)**, pp. 13-14.
[16] S. Mauw, J. H. S. Verschuren, E. P. de Vink, "A formalization of anonymity and onion routing", Computer Security–ESORICS 2004, Springer Berlin Heidelberg, **(2004)**, pp. 109-124.
[17] M. G. Reed and P. F. Syverson, "Onion routing", Proceeding of AIPA, vol. 99, no. 1, **(1999)**.
[18] P. Dhungel, M. Steiner and I. Rimac, "Waiting for anonymity: Understanding delays in the Tor overlay", Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on IEEE, **(2010)**, pp. 1-4.
[19] A. Chaabane, P. Manils and M. A. Kaafar, "Digging into anonymous traffic: A deep analysis of the tor anonymizing network", Network and System Security (NSS), 2010 4th International Conference on IEEE, **(2010)**, pp. 167-174.
[20] L. Øverlier, P. Syverson, "Improving efficiency and simplicity of Tor circuit establishment and hidden services", Privacy Enhancing Technologies. Springer Berlin Heidelberg, **(2007)**, pp. 134-152.
[21] A. Panchenko and J. Renner, "Path selection metrics for performance-improved onion routing", Applications and the Internet, 2009, SAINT'09, Ninth Annual International Symposium on IEEE, **(2009)**, pp. 114-120.
[22] A. Johnson and P. Syverson, "More anonymous onion routing through trust", Computer Security Foundations Symposium, 2009. CSF'09, 22nd IEEE, **(2009)**, pp. 3-12.
[23] J. Barker, P. Hannay and P. Szewczyk, "Using traffic analysis to identify The Second Generation Onion Router", Embedded and Ubiquitous Computing (EUC), 2011 IFIP 9th International Conference on IEEE, **(2011)**, pp. 72-78.
[24] O. Landsiedel, A. Pimenidis and K. Wehrle, "Dynamic multipath onion routing in anonymous peer-to-peer overlay networks", Global Telecommunications Conference, 2007, GLOBECOM'07, IEEE, **(2007)**, pp. 64-69.

Authors

Tian-Bo Lu, he was born in Guizhou Province, China, 1977. He is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, China. His technical interests include information security and computer network.

You-Wen Wang, he was born in ShanXi Province, China, 1989. He is a graduate student in School of Software Engineering, ShanXi University, China. His technical interests include information and network security, anonymous communication.

Ling-Ling Zhao, she is a graduate student in School of Software Engineering, Beijing University of Posts and Telecommunications, China. Her technical interests include Cyber-Physical System and P2P network.

Yang Li, he was born in Hunan Province, China, 1978. He is a PhD and his technical interests include information security, distributed computing and P2P network.

Xiao-Yan Zhang, she was born in Shandong Province, China, 1973. She is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China. Her technical interests include software cost estimation and software process improvement.