An Improved Method to Build a Circuit of Tor Hidden Service

Jiawei Liang[a] and Yi Liu

Faculty of Computer, Guangdong University of Technology, Guangzhou, China

[a]garwey@126.com

Abstract: Tor hidden service allows the hidden service provider to provide TCP service without revealing its location, protecting the anonymity of both client and server at the same time. The length of the hidden service circuit is one time longer than normal Tor circuit. Although tor hidden service ensures the anonymity of both sides, the transmission is slow and there are more encryption and decryption because of longer circuit. Aiming at the disadvantages of tor hidden service, an improved method to build a hidden service circuit is proposed. Shorter circuit improves its transmission efficiency while multipath lowers the probability of being traffic analyzed. At the end we carried out an experiment with Shadow simulator and analyzed the improvement of performance and security the proposed method brought.

Keywords: Tor, Hidden Service, Anonymous Communication, Circuit Establishment.

1. Introduction

Deep web [1] is the pages that cannot be searched by normal search engines, which contain enormous information. So far the widest used anonymous communication system is Tor (the onion router) [2]. Tor encrypts the message from users in layers and passes it via several onion routers instead of sending it straightly to provide both way low latency anonymous communication [3].

Normal Tor circuit protects the anonymity of the client. To protect the anonymity of the server at the same time, Tor provides hidden service, which allows hidden service provider to provide TCP service without revealing the location of the server [4]. In this way it can protect the server from DDos attack. Although it has been 10 years since the hidden service has been published, the protocol has not been changed basically. Because of that, there are a lot of disadvantages need to be improved. The length of the hidden service circuit is double the length of normal Tor circuit. A circuit with 6 hops greatly increases the delay. And also, if there is one node that was not chosen properly, there is a probability that the low bandwidth node becomes the bottleneck of the circuit which lowers the efficiency of the entire circuit. With the increase of the network traffic, the problem of hidden service is becoming prominence [5].

[6] proposed a technique to build an anonymous channels with the hidden service circuit of Tor. This technique can improve the transmission efficiency of the circuit and enhance the anonymity of both sides. But it is more vulnerable to traffic analysis attack compared to the original protocol.

[7] proposed a method to enhance traffic analysis resistance of hidden services by adding multiple path in a circuit, which is more robust under traffic analysis attack and more secure than the original hidden service. But the circuit structure is more complicated and there is not obvious improvement of communication efficiency.

Tor's hidden service has two major problems: low transmission efficiency and low security. This paper proposes an improved method aiming at these problems.

2. Tor Hidden Service

2.1 Work Flow of Tor Hidden Service

The most important part of Tor hidden service protocol is the rendezvous point, which transfers the data of both sides to make sure that each of them do not know the information of the other side. Meanwhile the rendezvous point does not know the identity of either side in order to ensure both sides' anonymity to the utmost [9]. The specific process of Tor hidden service protocol is introduced below.

Step 1: First the hidden server generates a pair of keys for the hidden service. Then it picks 3 to 10 relays and asks them to be the introduction point. The introduction points send back RELAY_COMMAND_INTRO_ESTABLISHED after verifying.

Step 2: The hidden server generates a hidden service descriptor for the hidden service, includes the public key of the hidden service, secret-id-part, publication time, protocol version, list of introduction points and the signature of hidden service descriptor. Then the hidden server generates two replicas of the descriptor and uploads them to hidden service directory servers for users to search. So far, the hidden service is deployed.

Step 3: The client who is going to request for the hidden service has obtained the onion address z.onion out of band, which is calculated with the public key of the server. The client calculated the descriptor id with the onion address so that it can download the hidden service descriptor from the hidden service directory server, after which the client obtained the list of introduction points and there public keys.

Step 4: The client randomly chooses a relay and asks it the act as rendezvous point. Then the rendezvous point responds the client with RELAY_COMMAND_RENDEZVOUS_ESTABLISHED cell.

Step 5: The client connects to one of the introduction point though a Tor circuit and sends it a RELAY_COMMAND_INTRODUCE1 cell to ask for its service. The cell contains IP, port, key of the rendezvous point, rendezvous cookie and the first half of Diffie-Hellman handshake gx.

Step 6: The introduction point forwards the above information to the hidden server in a RELAY_COMMAND_INTRODUCE2 cell, after which it sends a RELAY_COMMAND_INTRODUCE_ACK cell to the client to instruct the client to close the connection to the introduction point.

Step 7: The hidden server builds a circuit to the rendezvous point and sends it a RELAY_COMMAND_RENDEZVOUS1 cell containing rendezvous cookie, the second half of Diffie-Hellman handshake gy and digest KH.

Step 8: The rendezvous point sends the client the rest of the command cell though RELAY_COMMAND_RENDEZVOUS2 cell after verifying the rendezvous cookie.

Step 9: After receiving the data from rendezvous point, the client finishes the Diffie-Hellman handshake after verifying the handshake digest KH. The client sends a RELAY_COMMAND_BEGIN cell to the hidden server though the new circuit in order to begin communication.

Both sides can communicate though hidden service circuit like a normal Tor circuit. The main difference is that the circuit consists of 6 hops with 3 of them chosen by the client and 3 of them chosen by the hidden server. The delay is larger because the circuit is twice longer than a normal circuit. Through the process of establishment, both the introduction point and the rendezvous point do not know about either identity or location of both sides, which ensures the anonymity of both sides.

2.2 Shortcomings

Tor hidden service can protect the anonymity of both service requester and service provider at the same time, but there are still shortcomings, which are slow circuit establishment and large transmission delay. There are 4 normal Tor circuit needed to be built in the process of hidden service circuit establishment, so the circuit establishment is slow. And also, the data transferred has to be passed through 6 hops which is a lot slower than normal 3 hops circuit. Moreover, the security provided by the hidden service circuit is limited. If the attacker is able to observe both sides effectively, there is a probability that both sides' identity is revealed.

3. Improved Method to Build a Circuit

3.1 Process of Building a Circuit

Based on the original protocol modifications were made as following:

(1) Before sending cells to the introduction point, the client chooses a relay (OR1) as entry node and establishes a connection with it.

(2) After choosing a relay (OR2) as middle node, the server sends an extend command through introduction point. After receiving the command cell, the client passes the cell to the entry node by modifying the circuit ID without modifying the payload to extend the circuit to the middle node.

(3) The server sends an extend cell to the client through introduction point, indicating the middle node to extend the circuit the exit node.

(4) The server extends the circuit to m-1 exit nodes by the same way where m is set by the server [11].

(5) After the circuit is built, the server sends a command cell to instruct the introduction point to close the connection with the client. After knowing that the introduction point closed the connection with the client, which means the circuit between client and server is built successfully, the client begins to request the server for service through the new circuit.

In Fig. 1, the two "server" stands for the same server, ORm stands for a set of exit nodes. The hidden server guides the client to build a circuit to the server itself. The exits of the circuit are m nodes chosen by the server. The request sent by the client travels through OR1, OR2 and one of the exit nodes to the server. The response of the server travels back through the circuit. It is not necessarily through the same exit node.

The specific process of building a hidden service circuit is as the following:

1) The server choses the introduction points and builds connection with them.

2) The server generates a hidden service descriptors for the hidden service and uploads them to the hidden service directory servers for users to search.

3) The client downloads the hidden service descriptor from hidden service directory server.

4) The client randomly chooses a relay (OR1) and connects to it. After establishing connection, it requests the server to establish a connection by sending a RELAY_COMMAND_INTRODUCE1 cell to the introduction point.

5) The introduction point checks the cell and forwards the payload to the server in a RELAY_COMMAND_INTRODUCE2 cell.

6) The server chooses a relay as OR2 and sends the client a RELAY_EXTEND cell through the introduction point. After modifying the circuit id of the cell header, the client relays it to OR1 to extend the circuit to OR2.

7) The server chooses a relay as an exit node and sends OR2 a RELAY_EXTEND cell through the client to instruct the OR2 establish a connection to the exit node. A RELAY_EXTENDED cell

will be sent back after the connection is built successfully. The server sends the exit node a RELAY_MULTIPATH cell, containing a tunnel identifier (TID), and the exit node responses with a RELAY_MULTIPATH_ACK cell. So far, the main circuit is built. Next the auxiliary circuits begin to be built.

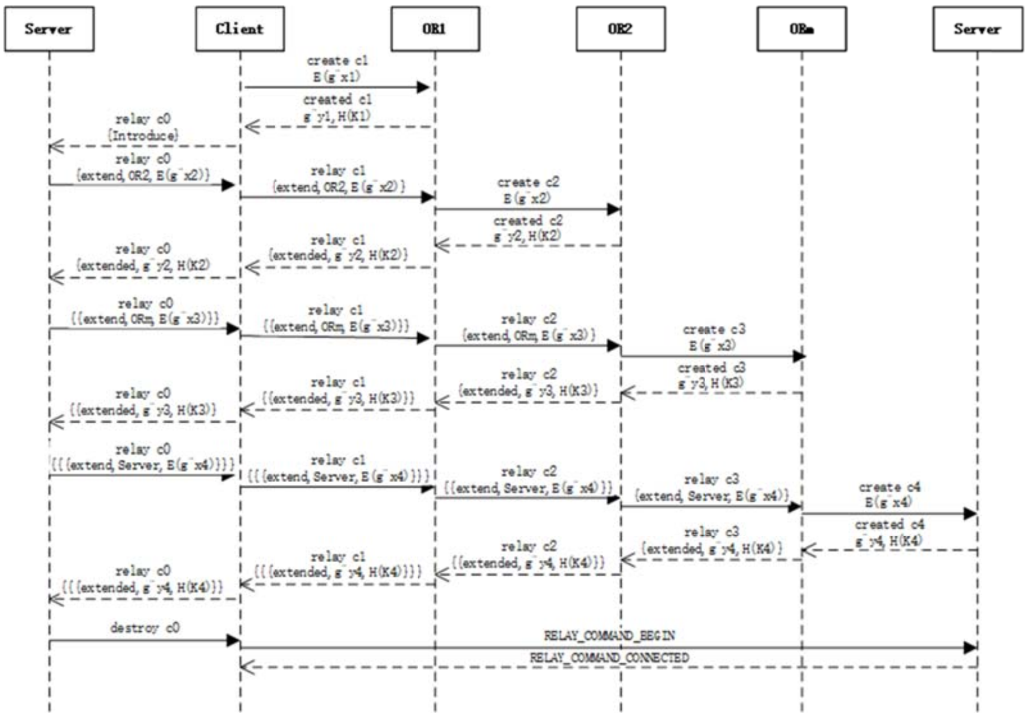


Fig. 1 Process of Building a Circuit

8) The server instructs OR2 to establish connections to the exit nodes by the above method and sends them RELAY_JOIN cells containing TID. The exit nodes correlate together with the same TID to compose a multipath [8]. The exit nodes reply with RELAY_JOINED cells to indicate that they joined the multipath successfully.

9) Repeat the above step m-1 times to build a multipath consists of m nodes. The m here is set by the server. When the server received m-1 acknowledgement, it knows that the multipath circuit is built successfully.

10) The server instructs the introduction point to close the connection with the client. The introduction point sends a RELAY_COMMAND_INTRODUCE_ACK cell to the client to instruct the client to close the connection with the introduction point.

11) The client sends a RELAY_COMMAND_BEGIN cell to the server through the new circuit in order to begin communication.

12) The server replies with a RELAY_COMMAND_CONNECTED cell. The communication began.

The topology of the circuit is shown in the Fig. 2.

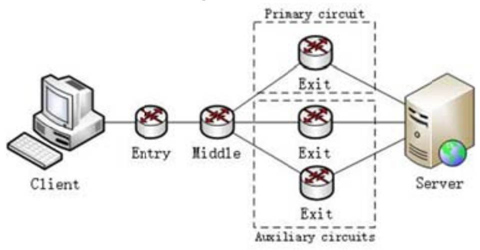


Fig. 2 Topology of the new circuit

3.2 Data Transmission

The data sent by the client travels through the entry node, the middle node and one of the exit nodes and arrived at the server eventually like what happens in a normal Tor circuit. What is different is that both the client and the server do not know about the entire information of the circuit. When the data sent by the client arrives at the middle node, the middle node relays it to a randomly chosen exit node which will relay it to the server. It is similar when the server sends data to the client. The server sends data to a randomly chosen exit node and the data will travels to the client along the circuit.

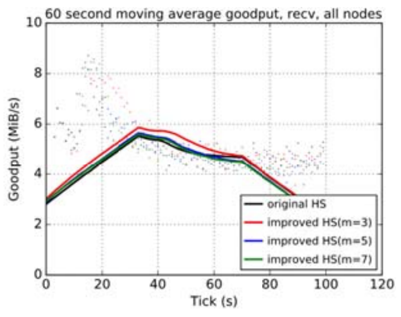
4. Experiment and Analysis

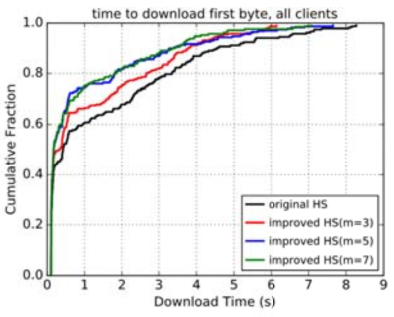
4.1 Experimental Design

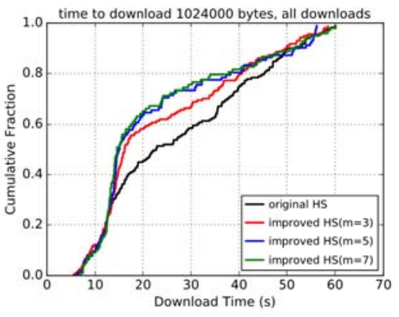
The tool used in the experiment is shadow, which is a simulator. It allows the user to simulate a Tor network. The user can set up the network topology, properties and behavior of the nodes in order to obtain the performance and parameters. There are many analysis tools in Shadow which can analyze the results of the experiments and generate graphs, from which the following graphs come.

The experiment environment is a computer with 8G RAM and i7-4790 CPU. It simulated a simplified Tor network, consisting of 20 servers, 50 web clients and 20 bulk clients. First, a simulation of the original protocol is carried out. Then we simulated the circuit built by the method we proposed, in which the values of m are set to 3, 5 and 7.

4.2 Experiment Results







(a) (b) (c)

Fig. 3 the simulation results of the improved method

In Fig. 3(a), black line stands for the original hidden service circuit, red line stands for the improved circuit where m equals 3 and blue line stands for the improved circuit where m equals 5. From the figure it can be concludes that the average goodput of the improved circuit is larger than the original circuit. And the average goodput of the circuit where m equals 3 is larger than the one with m equals 5. When m equals 3, the average goodput of the circuit increases from 7.54% to 12.52% compared to the original hidden service circuit while it increases from 4.09% to 7.40% when m equals 5. When m equals 7, it increases from 2.59 to 6.07%.

Fig. 3(b) shows the time to download first byte of different circuit. When m equals 3, the speed of downloading first byte is increased from 6.88% to 12.42% compared with the original hidden service circuit. When m equals 5, it increased from 10.63% to 26.32%. Time to download first byte of the improved circuit is shorter than the original circuit, which means quicker response. There will be more obvious performance optimization in the improved circuit under frequent request. When m equals 7, it increases from 6.45% to 18.65%.

Fig. 3(c) shows the time to download 1000MiB. When m equals 3, the time to download 1000MiB of the circuit increases from 13.31% to 28.32% compared with the original hidden service circuit. It increases from 25.75% to 51.68% when m equals 5. When m equals 7, it increases from 27.61% to 55.42%. The result shows that the transmission efficiency of improved circuit is higher and the circuit where m equals 7 is higher than the one with m equals 3. Because of more exit nodes, there will be less data needed to be transferred by each node, which reduces the potential congestion.

The new circuit is much shorter than the original circuit, which means faster transmission speed, shorter time to build a circuit and less encryption and decryption due to less relays. Exit point was changed to a set of nodes which share the traffic and speed up the transmission and response.

4.3 Security Analysis

Although the hidden service provided by Tor can ensure the anonymity of client and server to some extent, the anonymity will decline under traffic analysis [13]. When all the nodes in the multipath are adversaries, the circuit is dangerous and the relationship between two sides can easily be revealed. The probability of this happening can be calculated by the following formula:

$$P_{\text{compromised}}=\prod_{a=1}^{m} P(a) \tag{1}$$

m stands for the size of multipath. P(a) stands for the probability of the *ath* adversary relay being chosen, which is relevant to its bandwidth [14]. The higher the bandwidth is, the higher the probability of being chosen is. The value of P(a) can be calculated by the following formula:

$$P(a)= b_a /\sum_{j=1}^{n} bj \tag{2}$$

b_a here stands for the bandwidth of the *ath* adversary relay. n stands for the number of all nodes. It can be concluded from the above formula that the larger value of m, the lower probability of being compromised, and the higher security of the circuit.

There are over 6000 relays in Tor network, providing a bandwidth of over 25GiB/s [15]. We assume that the adversary can provide 100MiB/s bandwidth to acquire the relationship between the value of m and the probability of being compromised. The result is shown in table 1.

Table 1 Probability of the circuit being compromised

m	1	2	3	4	5
Pcompromised	0.39%	0.15%	0.06%	0.02%	0.01%

The original hidden service circuit does not have multipath, which is equivalent to the situation of the value of m is 1. At this time the probability of the circuit under traffic analysis attack is the highest. The value of m in method proposed in this paper is larger than 1, so the probability of the circuit under traffic analysis attack is lower than that of the original protocol. And the larger the value of m is, the higher the security of the circuit is.

5. Conclusion

In this paper we analyzed and researched Tor hidden service. And we introduced how Tor hidden service works, including the process to build a circuit. We analyzed the shortcomings of Tor hidden service. Aiming at the shortcomings, an improved method to build a circuit is proposed. The length of the circuit built by the method is only half of that of the original hidden service circuit. Multiple exit nodes enhance traffic analysis resistance of the circuit. At the end a simulation experiment is carried out with Shadow simulator. The result of the experiment shows that the method proposed in this paper is more efficient than the original hidden service. The security analysis shows that the circuit built by the proposed method has better resistance to traffic analysis, which brings higher security.

Acknowledgments

This work is supported by the national natural science foundation (No. 61572144), and the science and technology planning project of Guangdong Province (No. 2015B010128014, 2016B090918125).

References

[1] W. Liu, X. Meng, W. Meng, A Survey of Deep Web Data Integration. Chinese Journal of Computers (09), 1475-1489 (2007).
[2] Dingledine, R., Mathewson, N., Syverson, P., Tor: The second-generation onion router. Washington,DC: Navel Research Lab (2004).
[3] Tor Protocol Specification, https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt, last accessed 2018/01/03.
[4] Müller, K., Past, Present and Future of Tor Hidden Services. Høgskolen i Gjøviks rapportserie (2015).
[5] K. Bao, Research on Vulnerability Analysis for Dark Network Based on Tor. University of Electronic Science and Technology of China (2014).
[6] C. Wang, Research on Tor-based Construction Technique for Backward Anonymous Channels. Xidian University (2014).
[7] Yang L., Li F., Enhancing Traffic Analysis Resistance for Tor Hidden Services with Multipath Routing. In: Communications and Network Security IEEE, pp. 367-384 (2015).
[8] M. Chen, Research on Tor Content Classification Based on Traffic Analysis. Beijing Jiaotong University (2017).
[9] Tor Rendezvous Specification, https://gitweb.torproject.org/torspec.git/tree/rend-spec-v2.txt, last accessed 2018/01/03.
[10] C. Yi, Y. Zheng, A Mutiple-circuit Router Control Mechanism Based on Tor-A Way Protected Tor from end-to-end Timing Correlation Analyses. Computer Security (2010).
[11] L. Yang, F. Li, mTor: A multipath Tor routing beyond bandwidth throttling. In: Communications and Network Security. IEEE, pp. 479-487 (2015).
[12] Jansen, R., Hopper, N., Shadow: Running Tor in a Box for Accurate and Efficient Experimentation. MINNESOTA UNIV MINNEAPOLIS DEPT OF COMPUTER SCIENCE AND ENGINEERING (2012).
[13] Johnson, A., Wacek, C., Jansen, R., Sherr, M., & Syverson, P., Users get routed: traffic correlation on tor by realistic adversaries. In: ACM Sigsac Conference on Computer & Communications Security, pp. 337-348. (2013).
[14] W.Huang, Uniform Distribution Routing Algorithm Based on Tor Network. Shanghai Jiao Tong University (2012).
[15] The Metrics Portal, https://metrics.torproject.org/, last accessed 2018/1/10.