Removing Directory Servers from Anonymous Communication Systems using ID-Based Encryption to Improve Scalability

Hiroyuki Tanaka, Shoichi Saito, and Hiroshi Matsuo
Nagoya Institute of Technology
Gokiso-cho, Showa-ku, Nagoya, Aichi, 466-8555 Japan
tanaka@matlab.nitech.ac.jp, shoichi@nitech.ac.jp, matsuo@nitech.ac.jp

Abstract

Since the problem of disclosing personal information on the Internet continues to increase, many anonymous communication systems have been studied. Such systems usually use directory servers to manage public keys of participant nodes. However, this reduces anonymity because the query messages for the directory servers can give adversaries route information of anonymous communication channels. To solve this problem, applying ID-Based Encryption has been proposed, but in the existing method, directory servers continue to exist.

Our novel method, which can grasp assigned NodeIDs without directory servers, can be applied to existing DHT-based anonymous communication systems. Our proposal enhances scalability. This paper describes the structure of our proposed system and its application.

1. Introduction

As Internet proliferation continues, it provides more and more services. Some of these services need high confidentiality, for example, medical treatment and human rights counseling. Hence, research is being intensely investigated on anonymous communication systems to ensure anonymity on the Internet. Anonymous communication must satisfy the following three requirements [13]: a source node cannot be identified, a destination node cannot be identified, the data flow cannot be traced. These properties are collectively called *anonymity*, and communication with anonymity is called *anonymous communication*. Furthermore, communication channels with anonymous communication are called *anonymous communication channels*.

A typical technique to realize anonymous communication is a multistage relaying method that uses such multiple encryptions as Onion Routing [4, 18]. This method sends messages to a destination by multiple relay nodes, and a source multiple-encrypts messages using the public keys of relay nodes and the destination to conceal the final destination from all relay nodes. Relay nodes can only get the next-hop node after decrypting the messages.

The method requires a source to obtain the public keys of each relay node. For that purpose, Cashmere [22], Bifrost [7] and Bluemoon [15] need directory servers to manage the public keys. However, the method compromises anonymity because query messages for directory servers can reveal the routing information of anonymous communication channels to adversaries. This is obvious from research that infers client behavior from name resolution by DNS [5, 20]. To solve this problem, anonymous communication with ID-Based Encryption [2] has been proposed [6]. With IBE, communication for obtaining public keys can be omitted. However, sources must choose the NodeIDs of all relay nodes before building new anonymous communication channels. But the sources do not know how many nodes have already been joined and assigned NodeIDs. Hence, they cannot choose any NodeIDs for relay nodes. Because this Kate et al.'s proposal [6] requires directory servers to distribute assigned NodeIDs, it cannot remove directory servers from anonymous communication systems. We propose a novel method that can grasp the assigned NodeIDs without directory servers.

In Section 2, we describe the outline and the problem of existing anonymous communication systems. Section 3 introduces the details of our proposal. Section 4 describes how we apply it to existing DHT-based anonymous communication systems. Section 5 describes its implementation and performance analysis. Section 6 provides a conclusion.

2. Related work and its problems

This section outlines and describes the problems of existing anonymous communication systems.

2.1. Multistage relay using multiple encryptions

In IP communication, a source and a destination have to know each other's IP addresses for communication. Hence, in anonymous communication over IP, to conceal the source and the destination, messages must be relayed by many nodes [4, 18].

No relay node can identify a source and a destination to protect anonymity with the multistage relay. However, messages must obviously be delivered. Realization of the above requires that each relay node only learn the two IP addresses located before and after it on the anonymous communication channel. Public key cryptography solves this problem. Sources encrypt each next-hop NodeID as routing information in the reverse order of the relaying. Relay nodes can only get their next-hop node. The relay nodes cannot identify other nodes, except themselves, the next-hop node, and the previous node on the channel. In the end, no relay nodes can learn the entire anonymous communication channel. The above outlines anonymous communication with multistage relay and multiple encryptions. This method is used by many anonymous communication systems [3, 4, 7, 9, 11, 21, 22].

2.2. Problems of node and public key management

To construct an anonymous communication channel using multiple encryptions, a source first arbitrarily chooses some relay nodes. Hence, the source must obtain their public keys. There are three methods for managing nodes and distributing public keys. First, a source obtains a list in advance of all participant nodes, which include an IP address and a public key, as in the case of Tor [3]. Second, all nodes are controlled with a Distributed Hash Table (DHT), whose public keys are managed by directory servers. A source obtains node information with a DHT and public keys with such directory servers as Bifrost [7], Cashmere [22], or Bluemoon [15]. Cashmere's off-line central authority (CA) and Bluemoon's official CA are equivalent to directory servers. Third, public keys are built by IBEs such as Pairing Based Onion Routing [6].

In the first method, a source can arbitrarily choose relay nodes from the list of all participant nodes. It does not need special communication to get the relay node list. Consequently, relay nodes cannot be inferred by communication to get the list. A source can also obtain public keys with the list without extra communication. However, a source periodically gets a large list in this method. Therefore, it needs to pay a large cost to maintain the list.

In the second method, a source only chooses arbitrary DHT-IDs, and nodes managing them become relay nodes. In this method, a source does not need to look up relay nodes because each relay node looks up the IP address of the next-hop node using the next-hop DHT-ID in a relaying message. The method does not require communication to obtain candidates of relay nodes. However, a source needs to look up public keys in directory servers. Adversaries can surreptitiously discover relay nodes by monitoring query messages from the source to the directory servers.

In the third method, a source can arbitrarily choose relay nodes as in the first method because both methods need directory servers to distribute the list of participant nodes. Therefore, this method also needs to pay a large cost to keep the list. Moreover, adversaries can learn all nodes easily, and participant anonymity suffers.

3. Proposed method

We propose a novel method that can grasp assigned NodeIDs without directory servers and apply it with IBE to existing DHT-based anonymous communication systems. Our proposal enhances scalability.

This section outlines IBE and shows three new problems for applying IBE to DHT-based anonymous communication systems. We also propose a new introducing method.

3.1. ID-Based Encryption

IBE is a kind of public-key cryptography that derives a public key from an ID. IBE encrypts with a hash function and a destination's ID called an IBE-ID, which can be an arbitrary string. The hash function is shared by all users. IBE has a Private Key Generator (PKG), which we assume is operated by a trusted third party. PKG generates the master private key and a hash function, which it disseminates to all nodes. PKG also generates a private key of each node using the master private key and each IBE-ID. For IBE, a destination's public key does not need verification because it is based on the reliability of an IBE-ID and the hash function. In addition, Boneh and Franklin [2] suggest using a distributed PKG to realize a robust system.

3.2. Introducing IBE

Introducing IBE can prevent anonymity from being reduced by monitoring query messages for directory servers to solve the problems defined in 2.2. Regarding a NodeID as an IBE-ID can retrench the communication for obtaining a public key. Hence, we can achieve a robust anonymous communication method. Our proposal can be applied to various DHT-based anonymous communication systems.

3.3. Problems of introducing IBE

To introduce IBE into a DHT-based communication system and realize a system that does not depend on directory servers, we need to solve the following challenges. Our proposal, which solves these challenges, is described in detail in the following sections.

Unique NodeID assignment With IBE, PKG generates and assigns a private key corresponding to a NodeID as an IBE-ID. If a NodeID overlaps, some nodes have identical private keys and can decrypt the same message. Therefore, a collision-free NodeID assignment method is needed.

Verification of assigned NodeIDs Sources can even adopt arbitrary NodeIDs as relay nodes that have not been assigned yet in a system and encrypt messages with those NodeIDs as IBE-IDs. IBEs can create public keys that are not corresponding nodes, and no node can decrypt encrypted messages with the public keys. Hence, since the messages must be encrypted only using previously assigned NodeIDs, a new problem occurs: *how to get the previously assigned NodeIDs*. If a source communicates to verify previously assigned NodeIDs, anonymity is reduced. Therefore, we need a verification method for previously assigned NodeIDs that does not rely on communication.

Retrieval of a destination's NodeID In the NodeID assignment of anonymous communication systems, a NodeID is mainly a hash value of an IP address or a random value. The reasons for a random value for a NodeID are to uniformly assign the NodeIDs in DHT's ID space [9] and to prevent adversaries from choosing the NodeIDs they want [7, 10]. In addition, some systems [15, 22] use a random value for ID assignment, but they do not describe why. If NodeIDs are assigned randomly, we need to retrieve a destination's NodeID before beginning communication. But a source cannot communicate to retrieve the destination's NodeID for the same reason as the retrieval of public keys.

3.4. Unique NodeID assignment

An IBE-based system must not allow duplication of identical NodeIDs because a public key is derived from the NodeID as an IBE-ID. Therefore, we introduce a NodeID Allocator (NIA) that assigns one NodeID to just one node when the node joins an anonymous communication system. We assume that NIA is operated by a trusted third party.

3.5. Verification of previously assigned NodeIDs

To encrypt messages using only assigned NodeIDs and to protect anonymity, obtaining assigned NodeIDs needs no communication. We propose a rule that assigns NodeIDs to infer the assigned NodeIDs from a DHT routing table.

Such a simple ascending order NodeID assignment as 0,1,⋯ does not allow nodes to be sought effectively because these NodeIDs are not equalized in a DHT's ID space. Moreover, nodes cannot know the latest assigned NodeID because the latest node is not necessarily on their routing table. Therefore, we introduce *JoinNumbers*, which are assigned in ascending order to nodes. Reversing a JoinNumber in order of the bits is a NodeID (e.g., in case of 4 bits, 0001→1000, 0010→0100). The searching efficiency with DHT is not reduced because these NodeIDs are evenly spread over the DHT's ID space. Moreover, all new NodeIDs with this method are the middles of adjacent nodes at all times. Hence, all nodes can know the latest state of the NodeID assignment by reversing the next NodeIDs.

But since this method does not have a random nature, we add a partially random nature to it. We call a set of nodes whose JoinNumber can be found from 2^{i-1} to $2^{i}-1 (i \geq 1)$ "the ith *JoinGroup*", and randomly assign JoinNumbers in each JoinGroup. The 0th JoinGroup has only one node of NodeID 0. The next nodes also belong to the latest JoinGroup in our proposed method. Therefore, all nodes can know the current assignment state. Consequently, the proposed method provides communication-less verification of previously assigned NodeIDs and random NodeID assignment.

3.6. Retrieval of destination's NodeID

To send messages, sources need to know the destination's NodeID. But it is assigned randomly. We suppose that destinations open their own *Service Names* (SN) in place of their own NodeIDs to the public. Hence, sources can search for destinations by their SNs.

A simple discovery way is to flood the destination's NodeIDs and SNs. However, flooding is needed whenever a destination's NodeID is renewed. Furthermore, this method is not scalable. Another way is an existing secure service discovery method [19]. In this method, because search keywords are clear text, a third person can learn the following: what service is being provided and searched for by any nodes. Introduction Point was proposed by [12] to contact hidden servers. Introduction points are nodes that arrange to meet sources and destinations and can respond to the changes of the NodeIDs of destinations. An introduction point and a destination share a one-on-one relationship. Therefore, an introduction point can learn the related destination. We believe the relationship reduces anonymity.

for $i = 1$ to 160 **do**
  for $j = 2^{i-1}$ to $2^i - 1$ **do**
    repeat
      $JoinNumber = (random \bmod 2^{i-1}) + 2^{i-1}$
    until $JoinNumber$ is not already assigned
    $NodeID = bit\_order\_reverse(JoinNumber)$
  end for
end for

Figure 1: Pseudo code of NodeID assignment method

We enhance Introduction Point to increase anonymity and call it a *Multi-connected Introduction Point* (MIP). A node managing a hash value of an SN is an MIP of the SN. MIPs simultaneously maintain connections to many destinations and relay encrypted ID retrieval messages from sources to all connected destinations. Hence, many destinations that share the same MIP will receive the same message that was encrypted by an SN as a public key of IBE, and just one destination can decrypt and receive it. To keep multiple-connections to one MIP, a number of MIPs need to be under a number of destinations. Hence, nodes belonging to a small number of JoinGroups can only be an MIP. The number of JoinGroups for MIPs depends on the number of destinations in a system.

4. Application of proposal to existing systems

In this section, we apply our proposal to existing DHT-based anonymous communication systems.

4.1. Join protocol of proposal

To apply our proposal to DHT-based anonymous communication systems, we need to introduce an NIA and a PKG. All participant nodes assume that NIA and PKG can be trusted. Therefore, they can be combined into one. The join-protocol consists of the following four steps:

1. A joining node sends an assignment request of NodeID to the NIA when the node enters the system.
2. The NIA assigns a new NodeID to the node.
3. The node notifies the PKG about the assigned NodeID.
4. The PKG generates the corresponding private key and sends it and the hash function.

These communications are done over SSL.

4.2. NodeID assignment method

This section describes our proposed NodeID assignment method over Chord [17] and Pastry [16] as base DHTs. Chord is used in SALSA [11] and Bifrost. Pastry is used in Cashmere. Our proposal assigns NodeIDs to infer previously assigned NodeIDs from such known information as a DHT routing table. The NodeID assignment method is shown in Fig. 1. For Chord and Pastry, the ID space sizes are 2^{160}. As an initial condition, we consider that just one node (NodeID:0) has participated in the system. NIA assigns a NodeID from the 1st JoinGroup. In this method, if a node belonging to the xth JoinGroup is in its own routing table, which is the Finger Table of Chord and the Leaf Set, the Routing Table and the Neighborhood Set of Pastry, the previous assigned JoinGroups prove to be $(x-1)$th and under. Consequently, sources choose relay nodes from nodes belonging to the $(x-1)$th and under JoinGroups. We need to consider separately an assignment method for other DHTs, for example, Kademlia [8], which is used in Torsk [9].

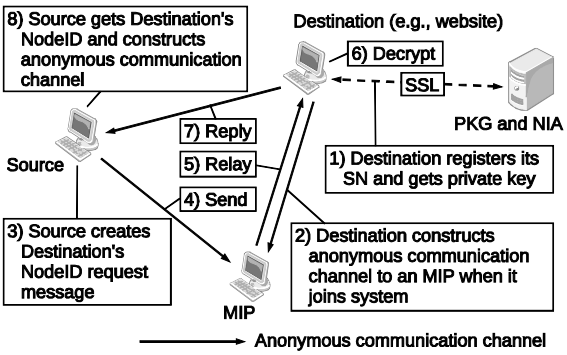

Figure 2: Retrieval of destination's NodeID using MIP

4.3. Retrieval of destination's NodeID

Figure 2 shows a proposed NodeID retrieval protocol using MIP for Bifrost. The following are its details in Bifrost:

Step 1) A destination registers its SN with PKG and obtains a private key corresponding to its SN as a public key.

Step 2) The destination constructs an anonymous communication channel between itself and the MIP in charge of the hash value of the SN.

Step 3) When a source connects to the destination, it generates a NodeID retrieval request message for the destination. The message is multiple encrypted using some relay nodes' NodeIDs and the destination's SN as public keys. Furthermore, it contains a reply header that is used to construct a homeward anonymous communication channel. The following is the request message's REQ structure:

$$REQ=P_{SN}(H_{DtoS}|IdR|S_D),$$

where "|" is union, P_X is a public key of IBE-ID:X, $P_X()$ is encryption with public key P_X, H_{XtoY} is a

header of a multistage encrypted message from node X to node Y, IdR is an ID retrieval command, $S_X$is a shared key between node X and the source, $S_X()$ is encryption with shared key S_X, D is the destination, and S is the source. At this time, since the source does not know the destination's NodeID, it cannot send the message to the destination.

Step 4) The source sends the message to the MIP managing the hash value of the SN. Message header H_{StoMIP} , which has route information of an anonymous communication channel between the source and the MIP, is encrypted as follows:

$$H_{StoMIP}=P_{T_1}(N_{T_1}|S_{T_1}|P_{T_2}(...P_{T_m}(\phi|S_{T_m}|\phi))),$$

where notes from T_1 to T_m are relay nodes, N_X is the next-hop NodeID of node X, and ϕ is null. When each relay nodes successfully decrypts the header using its own private key, it can obtain a shared key and a next-hop node. In addition, message body B_{StoMIP}, which is sent from the source to the MIP, is encrypted as follows:

$$B_{StoMIP}=S_{T_1}(...S_{MIP}(REQ)).$$

B_{StoMIP} is encrypted using shared keys that are enclosed in the corresponding header. The source can locate the MIP anywhere in the channel. It is not necessary that the source locates the MIP at the end of the channel.

Step 5) The relayed message goes through the channels built at Step 2 and reaches the destination and nodes.

Step 6) The destination and the nodes sharing the same MIP receive the message and decrypt it using their private keys corresponding to their own SNs.

Step 7) The destination, which can completely decrypt the message, sends a reply with its own NodeID to the source using an enclosed reply header in the received message. Reply header H_{DtoS} is created by the source and carried by the received message. Header H_{DtoS} is encrypted as follows:

$$H_{DtoS}=P_{F_1}(N_{F_1}|S_{F_1}|P_{F_2}(...P_{F_n}(\phi|S_{F_n}|\phi))),$$

where notes from F_1 to F_n are relay nodes on the reply channel. The source, which can be located anywhere in the reply route, receives the reply message through a return channel and gets the destination's NodeID. The NodeID of the destination is enclosed in a reply body. Body B_{DtoS} is encrypted and relayed to the source. Eventually, B_{DtoS}, which has reached the source, is the following:

$$B_{DtoS}=S_{F_1}(...S_D(ID_D)).$$

Note that B_{DtoS} is encrypted on the relay nodes that successfully decrypted the corresponding header. In this way, the destination never learns about the shared keys used on the reply route.

Step 8) The source completely decrypts the body and obtains the destination's NodeID. Then it constructs an anonymous communication channel between the source and the destination using the destination's NodeID.

Because the source, the MIP, and the destination are connected by anonymous communication channels, they cannot identify each other. Moreover, because the MIP forwards a NodeID request message to two or more destinations, it cannot identify which destination is the true destination of the request message. Hence, the source can retrieve the destination's NodeID without reducing anonymity.

4.4. Applications to DHT-based anonymous communication systems

To apply our proposal to DHT-based anonymous communication systems, the system needs to add PKG and NIA. Nodes are needed to add the following change and mechanisms:

- Changing join-protocol to the proposal;
- MIP mechanism;
- Requesting NodeID mechanism for sources;
- Providing NodeID mechanism for destinations.

The procedures after retrieving the destination's NodeID are not changed.

5. Implementation and evaluation

We implemented the proposal and measured its execution time for NodeID retrieval using MIP, which is assumed to be a challenge posed by the introduction of IBE. We combined Bifrost and the IBE library [1].

First, we successfully confirmed that anonymous communication can be realized without directory servers. In the measurement, we evaluated in two environments. The first was a local network system connected by 1G bps networks, and the second was a PlanetLab [14] connected all over the world. In both cases, 32 computers were joined to the environment systems. NodeID request messages go through the three channels: the first is from a source to an MIP (Step 4 in section 4.3), the second is from the MIP to a destination (Step 5), and the third is from the destination to the source (Step 7). Each channel has six relay nodes: 18 relay nodes in total.

The results of the NodeID retrieval times are 0.7 sec in the LAN and 7.0 sec in the PlanetLab. The first and second channels occupied over 40 % of each channel in both environments because these two channels are constructed when the request messages are conveyed. For practical use, this retrieval time is considerable. Hence reduction of channel construction time is a critical future issue.

6. Conclusion

In this paper, we proposed a novel method that can grasp previously assigned NodeIDs without directory servers. This method enhances scalability. In addition, we applied our proposed method to existing DHT-based anonymous communication systems and realized a system that does not need directory servers. Furthermore, we described the behavior of existing anonymous communication systems by applying our proposal not only to Bifrost but also to many other DHT-based anonymous communication systems. However, robust anonymity is realized in exchange for the high cost of NodeID retrieval processing. In the future, we will address the excessive retrieval time of destination NodeIDs in Internet environments. Moreover, remaining challenges are measures against dynamic joining and the leaving of nodes.

Acknowledgements

This research is partly supported by the Grant-in-Aid for Scientific Research (#20500064) from Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan.

References

[1] BAO Yiyang. ibe-javapairing. http://en.sourceforge.jp/projects/sfnet_ibe-javapairing/.

[2] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[3] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of 13th USENIX Security Symposium*, pages 303–320, 2004.

[4] D. Goldschang, M. Reed, and P. Syverson. Onion Routing for Anonymous and Private Internet Connections. *ACM SIGCOMM Computer Communication Review*, 42(2):39–41, 1999.

[5] K. Ishibashi, T. Toyono, and K. Toyama. Detecting Mass-Mailing Worm Infected Hosts by Mining DNS Traffic Data. In *ACM SIGCOMM workshop on Mining network data*, pages 159–164, 2005.

[6] A. Kate, G. M. Zaverucha, and I. Goldberg. Pairing-Based Onion Routing. *In Proceedings of 7th Privacy Enhancing Technologies*, pages 95–112, 2007.

[7] M. Kondo, S. Saito, K. Ishiguro, H. Tanaka, and H. Matsuo. Bifrost: A Novel Anonymous Communication System with DHT. In *Second International Workshop on Reliability, Availability, and Security*, pages 324–329, 2009.

[8] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. *Peer-to-Peer Systems*, pages 53–65, 2002.

[9] J. McLachlan, A. Tran, N. Hopper, and Y. Kim. Scalable Onion Routing with Torsk. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 590–599, 2009.

[10] P. Mittal and N. Borisov. ShadowWalker: Peer-to-peer Anonymous Communication Using Redundant Structured Topologies. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 161–172, 2009.

[11] A. Nambiar and M. Wright. Salsa: A Structured Approach to Large-Scale Anonymity. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 17–26, 2006.

[12] L. Øverlier and P. Syverson. Locating Hidden Servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 100–114, 2006.

[13] A. Pfitzmann and M. Waidner. Networks without user observability. *Computers & Security*, 6(2):158–166, 1987.

[14] PlanetLab. http://www.planet-lab.org/.

[15] K. P. N. Puttaswamy, A. Sala, C. Wilson, and B. Y. Zhao. Protecting Anonymity in Dynamic Peer-to-Peer Networks. In *IEEE International Conference on Network Protocols*, pages 104–113, 2008.

[16] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the Middleware 2001 : IFIP/ACM International Conference on Distributed Systems Platforms*, pages 329–350, 2001.

[17] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord : A Scalable Peer–To–Peer Lookup Service for Internet Applications. In *Proceedings of the 2001 SIGCOMM conference*, pages 149–160, 2001.

[18] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous Connections and Onion routing. *IEEE Journal on Specific Areas in Communications*, 16(4):482–494, 1998.

[19] S. Trabelsi and Y. Roudier. Secure Service Discovery with Distributed Registries. In *2nd IEEE Workshop on Service Discovery and Composition in Ubiquitous and Pervasive Environments*, pages 1–6, 2008.

[20] D. Whyte, E. Kranakis, and P. van Oorschot. DNS-based Detection of Scanning Worms in an Enterprise Network. In *Proceedings of the 12th Annual Network and Distributed System Security Symposium*, 2005.

[21] Y. Zhu and Y. Hu. TAP: A Novel Tunneling Approach for Anonymity in Structured P2P Systems. In *International Conference on Parallel Processing*, pages 21–28, 2004.

[22] L. Zhuang, F. Zhou, B. Y. Zhao, and A. Rowstron. Cashmere: Resilient Anonymous Routing. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design and Implementation*, pages 301–314, 2005.