Rendezvous Tunnel for Anonymous Publishing *

Ofer Hermoni
Department of Information Systems Engineering
Ben-Gurion University
Beer-Sheva, 84105, Israel
oferher@bgu.ac.il

Niv Gilboa
Department of Computer Science
Ben-Gurion University
Beer-Sheva, 84105, Israel
niv.gilboa@gmail.com

Eyal Felstaine
Department of Information Systems Engineering
Ben-Gurion University
Beer-Sheva, 84105, Israel
eyalfe@bgu.ac.il

Yuval Elovici
Dept. of Information Systems Eng. and Deutsche Telekom Labs at Ben-Gurion University
Beer-Sheva, 84105, Israel
elovici@bgu.ac.il

Shlomi Dolev
Department of Computer Science
Ben-Gurion University
Beer-Sheva, 84105, Israel
dolev@cs.bgu.ac.il

ABSTRACT

Many anonymous peer-to-peer (P2P) file sharing systems have been proposed in recent years. One problem that remains open is how to protect the anonymity of *all* participating users, namely, reader, server and publisher. In this work we propose a novel solution for a P2P file sharing system. **Our solution provides overall anonymity to all participating users**.

Servers in our system store shares of documents, and each share is reached through a *rendezvous tunnel* between the server and an address given by a hash of the document's name. To publish a document, the publisher first divides the document into shares, for each share finds the address of the entrance to the tunnel by hashing the document's name. Next, the publisher uses anonymous communication to reach the entrance of the rendezvous tunnel. We then use a random walk and an anonymous key exchange scheme to set keys along the rendezvous tunnel. The publisher finishes by inserting the shares into the servers through the rendezvous tunnels. A reader wanting to retrieve the document operates in a similar manner. The reader finds the address of the entrance to the rendezvous tunnels by hashing the document's name. Then, the reader uses anonymous communication to reach the entrance of the tunnels, retrieves the shares anonymously and reconstructs the document.

The novelty of this work is threefold. First, we introduce an anonymous key exchange protocol secure against an honest but curious adversary. The anonymity of the protocol is proved on the basis of the Decisional Diffie Hellman (DDH) problem. Second, we propose two solutions to build the rendezvous tunnel: basic and advanced. The basic solution is straightforward, while the advanced solution is based on the key exchange protocol. In the advanced solution, the key exchange is done between the publisher and each user along the rendezvous tunnel. Third, the rendezvous tunnel is used as a building block for an anonymous P2P file sharing system that provides anonymity to **all** participating users.

Categories and Subject Descriptors: C.2.0 [COMPUTER-COMMUNICATION NETWORKS]: General - *Security and protection*

General Terms: Security

Keywords: Anonymity, Publisher Anonymity, Peer-to-Peer Networks

1. INTRODUCTION

On-line communication occupies a great part of the daily lives of many people. Peer-to-peer (P2P) systems have recently become more popular, as we use them to chat, talk, file share etc. Anonymous P2P systems have thus become an important area of research [1, 3, 4, 5, 7, 8, 13, 14] and implementation [6]. Anonymity in P2P file sharing means that an adversary cannot link participating users to the content they share. Note that an adversary can act as a user, and thus the users also have to remain hidden with respect to one another.

A weakness common to many existing anonymity solutions is that they do not provide (or even consider) anonymity for *all* participating users, namely, publisher, server and reader. Solutions such as those given in [3, 12, 15] provide anonymity to the reader but do not protect the server that stores the shared document. Publishing solutions such as [5, 9] provide publisher and reader anonymity, but do not protect the servers or the creator of an index.

In our solution, anonymity for all users is achieved by using three anonymity tunnels (Figure 1). Anonymity tunnels are widely used in theory and in practice, e.g., Tor [6] uses tunnels to provide anonymity. We use two types of anonymity tunnels. The publishing and reading tunnels are sender anonymity tunnels. A sender anonymity tunnel is designed to protect the anonymity of the message's sender. In a sender anonymity tunnel, the sender knows the identity of the recipient, and anonymity is achieved by using a Mix [3] based protocol. The rendezvous tunnel is quite dif-

*This Research was Partly Supported by the Ministry of Science and Technology (MOST), the Israel Internet Association (ISOC-IL), the Frankel Center for Computer Science at the Ben-Gurion University, Rita Altura Trust Chair in Computer Science and Deutsche Telekom Labs at BGU.
An extended version appears in Technical Report #10-05-2010, of the Department of Computer Science, Ben-Gurion University of the Negev.

CCS'10, October 4–8, 2010, Chicago, Illinois, USA.
ACM 978-1-4503-0244-9/10/10.

ferent. In this tunnel, the initiator of the tunnel does not know the identity of the user at the end of the tunnel. This tunnel is built by using a random walk. The reading and the publishing tunnels protect the anonymity of the reader and the publisher, respectively, whereas the rendezvous tunnel protects the anonymity of the server.

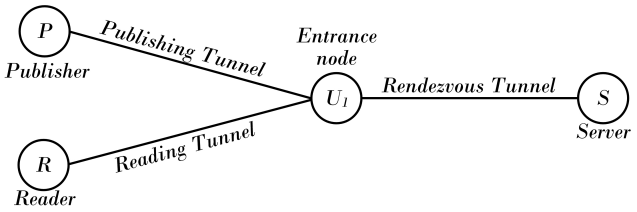

Figure 1: Three-tunnels system

2. SETTINGS AND REQUIREMENTS

Participants - In P2P file sharing networks, information is stored in units called *documents*. The participating users are *publisher*, *server* and *reader*. Retrieving a document in any system relies on *index mapping*. Index mapping enables a user to find the location of a specific document. Usually, the index mapping is stored in a database known as *index server*. Since the index constitutes a threat to users in P2P system, our system replaces a database mapping a document's name to a location with a publicly known hash function that provides this mapping.

Anonymity Model - According to the terminology of [10], anonymity means that a user is unidentifiable within a group of users, the *anonymity set*. Unlinkability of items (e.g., users, messages) means that an attacker cannot sufficiently distinguish whether these items are related or not. Sender–receiver unlinkability is provided against an adversary who is neither the sender nor the receiver.

Adversary Model - In anonymous P2P networks, the participating users, the server, reader and especially the publisher do not want their identity to be revealed. The adversary's goal is to link specific content to a participating user in the system and thereby to identify the user. Similarly to other schemes for anonymous P2P networks (e.g. Tor [6]), the adversary in our model is assumed to be *Honest but Curious*, which means that the adversary can control nodes in the network, but is obligated to follow the algorithms. We assume that the adversary can control at most t network nodes and the communication patterns do not reveal information. Hence, we do not deal with traffic analysis attacks. This adversary model is commonly used in anonymous P2P file sharing systems.

3. SOLUTION ARCHITECTURE

In this Section we propose two solutions. The basic solution is simple, however provides anonymity to all participating users. The advanced solution extends the basic solution. In addition to anonymity for all participating users, this solution provides a certain degree of traffic analysis resistance to the publisher. Moreover, the advanced solution conceals the content of the published document from the entrance node during the publishing phase. Note that documents are first divided into shares using an (n, k) IDA [11].

3.1 Basic Solution

There are basically two phases in this solution. First, in the *publication* phase, the publisher sends an encrypted share to the server. Next, in the *retrieval* phase, the reader retrieves the shares from the server and reconstructs the document. In the next two paragraphs, we explain the basic solution in detail.

Publication - First, by using a hash function on the name of the document concatenated to the number of the share, the publisher creates the index for all the shares of the document. The index of each share includes an entrance node - U_1, a reader seed - s_0 and an ID for the entrance node - ID_1. Then, the publisher encrypts the share sh with the reader seed obtaining $sh \oplus G(s_0)$. The publisher finishes by sending the encrypted share to the entrance node of the rendezvous tunnel via a sender anonymity tunnel. In order to build the rendezvous tunnel, the first node of the rendezvous tunnel (U_1) initiates a random walk (each user chooses randomly, among all network users, the next node in the tunnel) of length ℓ ($\ell > t$). During the construction of the tunnel, each user encrypts the document and forwards it to the server. To encrypt the document, each user creates randomly a session key (s_i) and XORs the received document with $G(s_i)$, where G is a pseudo random generator. When the message arrives at the server, the server saves the encrypted share $sh \oplus \bigoplus_{i=0}^{\ell} G(s_i)$.

Retrieval - The retrieval process is constructed from two messages - a query message and a response message. A reader, who wants to retrieve the document, creates the index mapping in the same way as the publisher. Note that the reader needs k shares to reconstruct the document. Then, the reader creates a sender anonymity tunnel, the reading tunnel, to U_1, and sends a query message to the server through the rendezvous tunnel and U_1. Each internal node along the rendezvous tunnel forwards the query message. When the server receives the query message, it sends the encrypted share back along the rendezvous tunnel. Each internal node XORs the message with $G(s_i)$ and forwards the message to the previous node along the rendezvous tunnel. The reader receives the response message, and decrypts the share with the reader seed. As soon as the reader accumulates k shares, the reader reconstructs the document.

Discussion on the Basic Solution - The basic solution provides anonymity to *all* participating users. The anonymity of the publisher and the reader is provided by the publishing tunnel and reading tunnel, respectively. The anonymity of the server is provided by the rendezvous tunnel. The main problem inherent in the basic solution is that the shares are sent to the entrance node U_1 encrypted with a key that is associated with the document's name. Hence, if U_1 is an active adversary, it can perform a brute force attack on all documents' names and find that U_1 is the entrance node of a specific share. The advanced solution deals with this drawback.

3.2 Advanced Solution

The advanced solution extends the basic solution with two main modifications. First, we use a new key exchange protocol to securely exchange keys between the publisher and nodes along the rendezvous tunnels. After the key exchange is complete, the publisher encrypts the shares with the exchanged symmetric keys and sends them to the servers

through the rendezvous tunnels. The second modification is that the publisher sends the encrypted shares through node U_2, the second node along the tunnel, and not through U_1. In this way, the brute force attack is not applicable during publication. Some time after the share is safe and securely encrypted in the server, the publisher binds U_1 to U_2 and the rest of the rendezvous tunnel.

Anonymous Key Exchange Protocol - The goal of the key exchange is to create symmetric keys in a way that two non-contiguous users along the tunnel will not know that they are taking part of the same key exchange. The key exchange protocol is designed under the assumption of an honest but curious adversary and is based on the Decision Diffie-Hellman (DDH) problem [2]. *Alice* exchanges keys with *Bob*, *Carol* and *David*, (Figure 2). We assume that each node knows only its immediate neighbors. To exchange keys, *Alice* sends a message containing three segments, with each segment destined to a different node. We now focus on the last segment sent from *Alice* to *David* (the other segments are built in the same way). *Alice* generates a random number R_3 ($R_3 \in_R q$), and calculates $g^{R_3} \bmod p$, where p is a large prime number, and q is another large prime such that $q|p-1$ and g is a generator of a multiplicative group of size q modulo p. *Alice* sends $g^{R_3} \bmod p$ to *Bob*. *Bob* selects a random number S_1, calculates $g^{R_3 \cdot S_1} \bmod p$ and forwards the message to *Carol*. *Carol* does the same as *Bob* and forwards the message to *David*. *David* as well selects a random number S_3 and calculates $g^{R_3 \cdot S_1 \cdot S_2 \cdot S_3} \bmod p$. The result is used as the symmetric key of *David* and *Alice*. *David* calculates and sends $g^{S_3} \bmod p$ to *Carol*. *Carol* calculates and sends $g^{S_3 \cdot S_2} \bmod p$ to *Bob*. *Bob* calculates and sends $g^{S_3 \cdot S_2 \cdot S_1} \bmod p$ to *Alice*. *Alice* calculates $g^{S_3 \cdot S_2 \cdot S_1 \cdot R_1} \bmod p$. Now only *David* and *Alice* know the symmetric key.

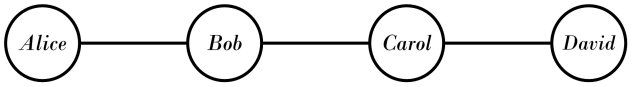

Figure 2: Alice exchanges symmetric keys

Publishing and Retrieving a Document - The overall flow of the advanced solution is as follows. First, similarly to the basic solution, the publisher uses hash functions over the document's name to build the index mapping. Then, for each share, the publisher selects at random the **second** node along the tunnel, U_2. Then we use random walk and the key exchange protocol to exchange keys between the publisher and each user along the rendezvous tunnel (excluding U_1). Then, the publisher encrypts the share and inserts it to the server through U_2. Each user along the tunnel removes the symmetric key by XORing the share with S_i. Then in a similar way to the basic solution, each user encrypts the document by XORing it with a new random key s_i. Later, after a while, the publisher binds U_1 to the tunnel. The retrieval process is done exactly in the same way as in the basic solution.

Fault Tolerance - Let us assume that nodes are expected to leave the network with probability P. If a node leaves the network, all the shares that this node is part of their rendezvous tunnels become unavailable. Hence, the probability that a tunnel is active is $P_t = (1-P)^{\ell}$, where ℓ is the tunnel length. Assuming that $N >> \ell \cdot n$ (where N is the number of nodes in the network and n is the number of shares created for each document), for simplicity we can say that with a good approximation there is no intersection between different tunnels. Since we use an (n,k) IDA, unless at least k out of n tunnels are active, the document is not available. Hence, the probability that the document is available is $\sum_{i=k}^{n} \binom{n}{i} {P_t}^i (1-P_t)^{n-i}$. With careful selection of n and k, we can achieve a high probability for document's availability.

4. CONCLUSION

In this work we introduced a new anonymous key exchange protocol under the assumption of an honest but curious adversary. We showed how to build a rendezvous tunnel. We presented two solutions to anonymous P2P networks. The first solution is simple, as it uses the rendezvous tunnel without key exchange. The second solution enhances the first solution by using the rendezvous tunnel with the symmetric key exchange protocol. Our solutions thus provide overall anonymity to **all** participating users.

5. REFERENCES

[1] A. Beimel and S. Dolev. Buses for anonymous message delivery. *Journal of Cryptology*, 16(1):25-39, 2003.

[2] D. Boneh. The decision diffie-hellman problem. In *Lecture Notes in Computer Science*, pages 48-63. Springer-Verlag, 1998.

[3] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.

[4] I. Clarke, O. Sandberg, B. Wiley and T. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Workshop on Design Issues in Anonymity and Unobservability*, pages 46-66, 2000.

[5] R. Dingledine, M. J. Freedman and D. Molnar. The free haven project: Distributed anonymous storage service. In*Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, Springer-Verlag, LNCS 2009, July 2000.

[6] R. Dingledine, N. Mathewson and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[7] S. Dolev and R. Ostrovsky. Xor-trees for efficient anonymous multicast and reception. *ACM Transactions on Information and System Security*, 3(2):63-84, May 2000.

[8] O. Hermoni, N. Gilboa, E. Felstaine and S. Shitrit. Deniability - an alibi for users in p2p networks, In *COMSWARE*, pages 310-317, 2008.

[9] A. R. Marc Waldman and L. Cranor. Publius: A robust, tamper-evident, censorship-resistant and source-anonymous web publishing system. In *Proceedings of the 9th USENIX Security Symposium*, pages 59-72, August 2000.

[10] A. Pfitzmann and M. Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. Aug. 2010. v0.34.

[11] M. O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM*, 36(2):335-348, 1989.

[12] M. K. Reiter and A. D. Rubin. Crowds: anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66-92, 1998.

[13] A. Serjantov. Anonymizing censorship resistant systems. In *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS 2002)*, March 2002.

[14] S. Shitrit, E. Felstaine, N. Gilboa, and O. Hermoni. Anonymity scheme for interactive p2p services. *Journal of Internet Technology*, 10:299-312, 2009.

[15] P. Syverson, D. Goldsclag, and M. Reed. Anonymous connections and onion routing. In *Proceedings of the IEEE 18th Annual Symposium on Security and Privacy*, pages 44-54, Oakland, California, 4-7 1997.