Bruit Bait: Peer To Peer Systems

[1]P.Devika, [2]Dr.R.S.Ponmagal
[1] PG scholar, Information Technology
[2]Professor/Dept of CSE
[1, 2]DR.M.G.R Educational and Research Institute
University Chennai-95 India

Abstract:
Anonymizing Peer-to-Peer (P2P) systems often incurs extra costs in terms of transfer efficiency; many systems try to mask the identities of their users for privacy considerations. Existing anonymity approaches are mainly path-based: peers have to pre-construct an anonymous path before transmission. The overhead of maintaining and updating such paths is significantly high. Bruit Bait, a lightweight mutual anonymity protocol for decentralized P2P systems is proposed in this project. Bruit Bait employs a random walk scheme which frees initiating peers from the heavy load of path construction. Compared with previous RSA-based anonymity approaches, Bruit Bait also takes advantage of lower cryptographic overhead by mainly utilizing a symmetric cryptographic algorithm to achieve anonymity.

I. Introduction

1. Peer-to-Peer

Peer to Peer networks, such as Napster, Gnutella, and Bit Torrent, have become essential media for in-formation dissemination and sharing over the Internet. Concerns about privacy, however, have grown with the rapid development of P2P systems. In distributed and decentralized P2P environments, the individual users cannot rely on a trusted and centralized authority, for example, a Certificate Authority (CA) center, for protecting their privacy. Without such trustworthy entities, the P2P users have to hide their identities and behaviors by themselves. Hence, the requirement for anonymity has become increasingly critical for both content requesters and providers.

A number of methods have been proposed to provide anonymity. Most, if not all, of them achieve anonymous message delivery via non traceable paths comprised of multiple proxies or middle agent peers known as path-based approaches, require users to setup anonymous paths before transmission. Although path-based protocols provide strong anonymity, an anonymous path has to be pre constructed, which requires the initiator to collect a large number of IP addresses and public keys. Also, an initiator has to perform asymmetric key based cryptographic encryptions, when wrapping the layer-encrypted packets. Both the peer collection and content encryption introduce high costs. Practically, users often expect to establish a long anonymous path and update the path periodically to defend against the analysis from attackers.

To address the above issues, we propose a non-path-based anonymous P2P protocol called Rumor Riding. In Rumor Riding, we first let an initiator encrypt the query message with a symmetric key, and then send the key and the cipher text to different neighbors. The key and the cipher texts take random walks separately in the system, where each walk is called a Rumor. Once a key Rumor and a cipher Rumor meet at some peer, the peer is able to recover the original query message and act as an agent to issue the query for the initiator. We call the agent peer as a sower in this paper. The similar idea is also employed during the query response, confirm, and file delivery processes. Thus, the Rumors serve as the primitives of this protocol to achieve mutual anonymity and meet the design objectives.

In RR, anonymous paths are automatically constructed via the Rumors' random walks. Neither the initiator nor the responder needs to be concerned with path construction and maintenance.

RR employs a symmetric cryptographic algorithm to achieve anonymity, which significantly reduces the cryptographic overhead for the initiator, the responder, and the middle nodes. In addition, as initiating peers have no requirement on extra information for constructing paths, the risk of information leakage, caused by links that are used for peers to request the IP addresses of anonymous proxies, is eliminated.

II. Bruit Bait

Bruit Bait includes five major components: Rumor Generation and Recovery, Query Issuance, Query Response, Query Confirm, and File Delivery.

2.1 Rumor Generation and Recovery

RR employs the AES algorithm to encrypt original messages. The key size is 128-bit. To determine whether a pair of cipher and key rumors hit, we employ a Cyclic Redundancy Check (CRC) function to attach a CRC value, CRC(M), to the message M. For received key rumors and cipher rumors, the sower S uses AES to recover a message 'M' and the checksum

CRC(M’). It then performs the CRC function to the recovered M’ and compares the result with CRC(M'). If they match, the sower S is aware that it has successfully recovered a message M.

2.2 Query Issuance

When an initiator wishes to issue an anonymous query, it first generates the query content q, and a public key K_I^+. Node I then uses an AES cryptographic algorithm to encrypt q into a cipher text C with a symmetric key K. It organizes the key K and the cipher text C into two query rumors, qK and qC, each packet is labeled with a Descriptor ID, a string that uniquely identifies the packet. RR also uses the descriptors to identify rumors. Thus, two random number strings, ID_{qK} and ID_{qC}, are used to label the two rumors. After generation, I forward the rumor messages to two randomly chosen neighbors, The query cipher rumor and the query key rumor then start their random walks.

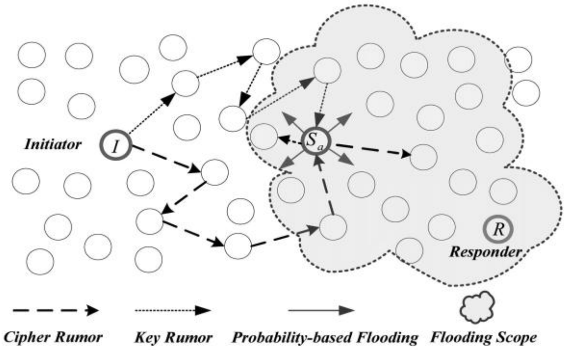


Fig.1. Query Issuance

RR requires every node to temporarily keep a local cache to store the received rumors. When a node receives a query key rumor, it performs the rumor recovery procedure to check all cached cipher rumors. If a decrypted rumor holds a plaintext matching the CRC value, q will be successfully recovered. Whatever there is a match or not, this intermediate node reduces the TTL value of the received rumor by one, keeps a temporary record containing the ID of the rumor in the local cache, and forwards it to a randomly chosen neighbor. The procedure continues until the TTL value of this rumor is reduced to zero. For the received query cipher rumor, the process is similar. Therefore, if a pair of query rumors reach a certain node, no matter what the sequence is, this node will eventually recover the original q. The key issue with this procedure is that the number of rumors and their initial TTL values need to be carefully selected so that at least one pair of rumors, including a key and a cipher, will meet. Thus, if an adversary receives a message with a Hops set as zero, it knows that the node sent the message is the initiator. To avoid this, RR initializes a nonzero positive number in the Hops fields of rumors before sending them out.

If one intermediate node that recovers q is willing to act as an agent peer, it conducts a search on behalf of the unknown I . We call this agent node a sower. When a peer identifies itself as a sower, it checks the TTL values defined in the rumors. If they are not zero, the sower forwards the rumors out, so that if there are attackers who can overhear some of the messages sent to the sower, it is still not trivial to determine whether or not the peer is a sower. Next the sower, Sa attaches the original query message q with its IP address, and then issues the query marked with a label IDq in a plaintext (IDq is also used for Sa to locate the correlated q_K and q_C). In this operation, we avoid a blind flooding. Instead, we employ a probability-based-flooding, in which the sower selects a subset of its neighbors and issues the query. Note that the sower does not send the query to the nodes which sent or have been sent the two rumors of this query. For the neighboring nodes that do not send or have not been sent the two rumors, the sower sends the plain text query to each of those nodes with a probability p. The p is like a threshold. The sower can first compute a random value between [0, 1], and then compare to the p. If the value is less than p, the sower sends the query, otherwise does not send. Upon receiving the query, the neighboring nodes forward the query to each of its neighbors (expect the sower) with the probability p. Such a procedure continues until the query packet exhausts its TTL. The selective flooding has a constrained flooding scope compared to the blind flooding, which can reduce the redundant traffic caused by multiple sowers’ flooding. The probability p is a systematic parameter.

3.3 Query Response

When a receiving node the query has a copy of the desired file, it becomes a responder R. To respond to the query, R encrypts the plain text of the response message r, using the initiator’s public key KI . It encrypts using AES, where K^R is the public key generated by R, and encloses the cipher text and the key into two response rumors

After being sent out from R, two rumors start their random walks in the system. RR guarantees that at least one pair of rumors meet at a certain peer Sb . We use lrK and lrC to denote their paths from R to Sb . Sb decrypts the cipher text in rC with the key in rK , and recovers the IP address of sower Sa .

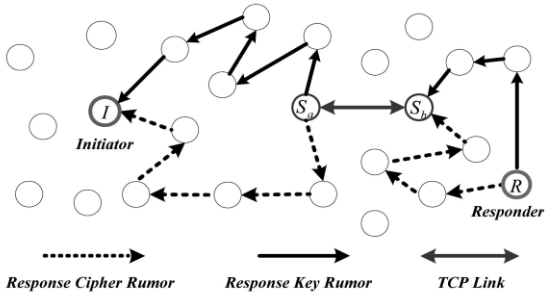


Fig.2. Query Response

If S_b volunteers to forward the response for R, it contacts S_a via a TCP connection, and forwards these two response rumors to S_a . Note that S_b also attaches its IP address, IDq, ID_{rK}, and ID_{rC} to the two rumors. When Sa receives the responses r_K and r_C , it delivers them to the originating peers of q_K and q_C . Two response rumors are marked with ID_{qK} and ID_{qC} , to help them walk along the reversed paths of l_{qK} and l_{qC}. The successor nodes continue this procedure. Thus, two response rumors make use of lqK and lqC to reach I .

3.4 Query Confirm

In the query confirm phase, I uses the responder's public key to encrypt the confirm message c. It then encrypts<(c)K_R^+; ID_{rK}; ID_{rC}; IP_{Sb}> and obtains two confirm rumors, c_K and c_C, which take random walks in the system. Note that two confirm rumors are marked with new descriptors: ID_{cK} and ID_{cC}. We assume that c_K and c_C collide in a new sower S0a. We denote their paths from I to S'$_a$ by l_{cK} and l_{cC} . When S0a recovers the IP address of S_b from c_K and c_C, it directly contacts S_b to forward c_K and c_C attached with ID_{rK} and ID_{rC} via a TCP link, as shown in Fig. 3. The c_K and c_C are then delivered along the reversed paths of l_{rK} and l_{rC} until they reach R.

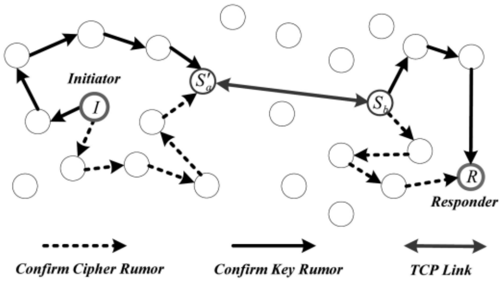


Fig.3. Query Confirm

3.5 File Delivery

After recovering the confirm message from (c)K_R^+ , using its private key K_R, R employs a digital envelope technique to encrypt the file into cipher CF . Instead of including CF into the rumor generation, R encrypts <ID_{cK}; ID_{cC}; IPS0a > to generate the data cipher rumor and the data key rumor, and attaches the digital envelop payload to the data cipher rumor. The large data cipher rumor and the small data key rumor first take random walks to meet each other at a sower S'$_b$, then traverse the path from S'$_b$ to S'$_a$ via a TCP connection, and eventually reach I along the reversed paths of l_{cK} and l_{cC} . Upon receiving the digital envelop, I recover the desired file using its private key. For large-size files, responders can split them into multiple segments.

IV. Discourse

We now examine several key issues in the RR design. We first focus on how to ensure that each query has at the least one sower and that the sowers are evenly distributed over the system. We then discuss the attack models and analyze the anonymity degree of RR.

4.1 Sower Distribution and Collision Rate

In RR, we select the random walk as the rumor spreading method. P2P systems mainly utilize three communication patterns to deliver messages: flooding, random walk, and end-to-end delivery. We select random walk as the fundamental anonymizing method. The distinct features of random walk mechanism are as follows: First, random walk mechanism introduces randomness to the message delivery such that the difficulty for attackers to trace back to the initiator or responder is increased. Second, this mechanism potentially involves all peers in the anonymizing process, so that the anonymous proxy set is extended from a small group in path-based approaches to the entire P2P network.

Specifically, RR rumors are sent in random directions, and each peer forwards a rumor to one of its neighbors without any bias.

The collision distance is another important factor balancing the tradeoff between the user anonymity and the query delay. Initiators hope that the sower peers reside as far away as possible, since the sowers recover the query messages and might help adversaries to locate the initiator if they are compromised. Thus, the number of rumors needs to be limited as well.

4.2 Anonymity Analysis

In this section, we first discuss the degree of anonymity that RR achieves, and then analyze the protocol effectiveness under various attack scenarios

.

4.2.1 Anonymity Model

There are two main categories of anonymity models for defining the anonymity degree. The models in the first category define the anonymity of a certain node as the number of peers that have an equi probable chance of being the given node, which is termed as anonymity set. The second category employs measurements based on information theory, to reflect the similarity between two entities, such as the input/output links or real/suspected participants.

4.2.2 Attacks

It claim that the protocol achieves unlink ability to the initiator and responder, if they cannot be identified when communicating with each other. In our attack model, we assume that, based on the records, the adversary nodes are able to observe and store the communication traversing them and guess the identity of nodes that initiated those transmissions.

Collaborating attack: In RR, a sower selects a subset of its neighbors to send the plaintext query, and the two collaborating nodes will not receive the query. In this way, adversaries only bet that the monitored node is an initiator or a responder. Hence, RR is not subject to the local collaborating attack, if the adversaries cannot compromise more than three neighbors of the monitored node.

Timing attack: In a timing attack, the adversary deduces the correlation among the timings of packets, such as the response time of a query, the time difference of a query, the time interval between two sequential packets, etc., is invulnerable in that 1) rumors are delivered over the overlay network in a random walk manner, and RTT measurements do not reveal the real distance to the responder; 2) if adversaries want to trace the rumor via the time difference to locate the responder, they need to trace one query rumor from the initiator to a sower, then trace the plaintext query message from the sower to the responder, which is not trivial; and 3) a sower issues a request only after it obtains a pair of query rumors, so the response time is mainly dependent on the random walks of rumors, which are unpredictable. All of these factors make it difficult to launch a timing attack.

Predecessor attack: In RR, rumors walk randomly and interact with random sowers. The sowers of a given initiator or responder are unpredictable and randomly distributed over the system. Hence, adversaries are not able to perform such an attack to identify the initiator or responder via sowers.

***Traffic analysis attack*:** An adversary can extract traffic flow information such as packet count, message volume, and communication pattern, etc., RR is much less vulnerable to this attack since subsequent messages do not belong to the same traffic, and there are not any continuous paths in RR.

***Traceback attack*:** Adversaries start from a known sower to trace back to the initiator along the rumor paths. The adversary examines the stored routing state of the peers to identify the paths between the initiator and responder. We consider the users' anonymity in two attack scenarios:

1) One-way back tracking: adversaries that are on the rumor path back-track and collaborate with each other to detect the source node of this rumor;
2) Multiple-ways back tracking: at least one adversary intercepts both the cipher rumor and the key rumor. RR achieves a high degree of anonymity under traceback attacks.

V Experiments

Additional latency of data delivery, bandwidth consumed by anonymous traffic, and crypto processing, if they exist, are necessary in order to provide anonymity.

5.1 Metrics

We use the following metrics for evaluating Bruit Bait:

Collision rate. To verify the theoretical results we examine the distribution of collision rate with real traces. Besides the verification, we also use the results to guide the selection of rumor parameters.

Collision distance. A longer collision distance often means a higher anonymity level, but also increases the delay of a query as well as the traffic overhead. On the other hand, the collision distance must be sufficiently large to guarantee sower diversity, as we discussed in Section 4.

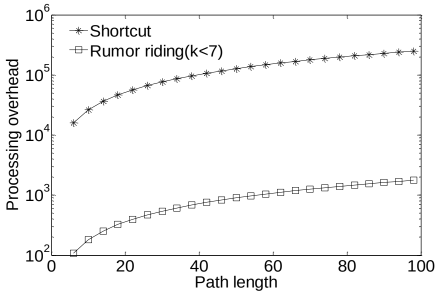
Sower diversity. The metric reflects the distribution of sower locations in the P2P systems. Evenly random distribution of sower location leads to a higher anonymity degree.

Number of sowers. Since each sower implements a selective flooding search for an initiator, too many sowers will incur a large number of replicated query messages, and too few sowers will result in failure on providing enough redundancy and reliability.

Traffic overhead. The amount of traffic overhead represents the comprehensive latency in data delivery and bandwidth. For each message enrolled in one query cycle, we calculate the sum of the distances that this message passes through. We define the extra traffic overhead as the total traffic overhead of a query cycle in an anonymized P2P system minus that of a query cycle in a nonanonymized P2P system.

Response time. In P2P systems, it is defined as the time elapsed from the start of rumor spreading to the time when the initiator receives the first response message.

crypto latency. The overhead incurred by the main cryptographic algorithms. We use the processing overhead in one AES operation as the basic unit to make conversions between RSA and AES.



VI. Conclusion

We propose a lightweight and non-path-based mutual anonymity protocol for P2P systems, Rumor Riding (RR). Employing a random walk concept, RR issues key rumors and cipher rumors separately, and expects that they meet in some random peers. The results of trace-driven simulations and simple implementations show that RR provides a high degree of anonymity and outperforms existing approaches in terms of reducing the traffic overhead and processing latency. We also discuss how RR can effectively defend against various attacks.

Future and ongoing work includes accelerating the query speed, introducing mimic traffic to confuse attackers, and optimizing the k and L combination to further reduce the traffic overhead. We will also investigate other security properties of RR, such as the unlinkability, information leakage, and failure tolerance when facing different attacks. It would also be interesting to explore the possibility of implementing this lightweight protocol in other distributed systems, such as grid systems and ad-hoc networks.

References

[1] A.Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: an approach to universal topology generation", In Proceedings of the International Workshop on MASCOTS) 2001..

[2] D. Chaum, "Untraceable electronic mail return addresses, and digital pseudonyms", Communications of the ACM, 1981.

[3] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing", Communications of the ACM, 1999.

[4] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications", In Proceedings of IEEE INFOCOM, 1999.

[5] M. K. Reiter and A. D. Rubin, "Crowds: anonymity for web transactions", ACM Transactions on Information and System Security, 1998.

[6] V. Scarlata, B. N. Levine, and C. Shields, "Responder anonymity and anonymous Peer-to-Peer file sharing", In Proceedings of IEEE ICNP, 2001