DDoS Attacks using P2P Networks

Pankaj Kohli and Umadevi Ganugula
Centre for Security Theory and Algorithmic Research
International Institute of Information Technology
Hyderabad, India - 500032
{pankaj_kohli, umadevi}@research.iiit.net

April 25, 2007

Abstract

P2P networks have been extensively used for file sharing. Without any form of authentication, messages are relayed between peers in a P2P network. In such a scenario, the P2P network could be exploited by an attacker to launch a DDoS attack against any arbitrary host on the Internet. This paper studies two such approaches that can be used to launch a DDoS attack in Overnet, which is one the largest P2P file sharing network. The first one involves poisoning the distributed file indexes, which leads to TCP Connection DDoS attack, and the second one involves poisoning the routing table of the peers, leading to Bandwidth Flooding DDoS attack. We first describe these two attacks in Overnet, which is one of the largest P2P file sharing network. We then propose a simple yet effective solution to defend against the routing table poisoning in the P2P file sharing networks.

Contents

1 Introduction

A peer-to-peer (or P2P) network relies primarily on the computing power and bandwidth of the participants in the network rather than concentrating it in a relatively low number of servers. P2P networks are typically used for connecting nodes via largely ad-hoc connections. Such networks are useful for many purposes. Sharing content files containing audio, video, data or anything in digital format is very common, and real-time data, such as telephony traffic, is also passed using P2P networks. With more than 10 million users simultaneously connected to Gnutella, Overnet, FastTrack, Kazaa and BitTorrent, P2P file sharing continues to be one of the most important applications in the Internet today. P2P applications today contribute more traffic than any other application on the Internet. With such a huge user base and lack of any authentication, P2P networks can be leveraged by an attacker to launch a DDoS attack against a victim machine on the Internet. The victim need not be a participant in the P2P network, and could be a web server, a mail server or even a home user's desktop.

P2P networks have indexes. An index in a P2P network is a set of mappings from keys to values. For a P2P file sharing network, the keys are file hashes and the values are locations at which the file corresponding to the file hash is present. Location is usually described by the tuple $\langle NodeID,\ IP : Port \rangle$, where NodeID uniquely identifies a peer in the P2P network. Thus, an index record in a P2P file sharing network specifies which file is present at which location. An attacker can exploit a P2P file sharing network in two ways to launch DDoS attacks - index poisoning and routing table poisoning. In an index poisoning attack, the attacker plants false index records in a large number of peers. These false index records indicate that a popular file is present with the victim. When any peer tries to search for that file, it receives this false index record from the poisoned peer. Since the file is popular, there will be large number of requests for that file. On receiving the false index record, the searching peers try to connect to the victim, trying to download the file, filling up the number of allowed connections, preventing legitimate users from connecting to the victim. In the routing table poisoning attack, the attacker makes the victim, a neighbor of a large number of peers by sending them false node announcement messages. Whenever a peer receives a search query or a maintenance message, it may select the victim from its routing table and forward the message to the victim. If the attacker poisons the routing table of a large number of peers, the victim may receive a flood of search queries and maintenance messages, saturating the victim's link.

This paper is organized as follows. We first describe Overnet, which is the largest deployed DHT till date. We then describe how an attacker can exploit Overnet to launch DDoS attacks. In section 2, we describe the Reliable Index Exchange Protocol (RIEP), which is the only known solution till now for index poisoning attack. We also discuss some related issues in P2P network security. In section 3, we describe our approach for defending against routing table poisoning attack, which authenticates a peer with just two messages, one in each direction, without using PKI. In section 4, we describe implementation details of RIEP. In section 5, we discuss future work and unsolved issues in P2P network security.

1.1 Overnet

Overnet is claimed to be the largest deployed DHT till date, with more than one million peers alive at any given time of the day. Many popular P2P file sharing clients, such as eDonkey2000 and eMule run over Overnet. Overnet is built on top of Kademlia DHT. All participant nodes have equal roles and no hierarchy exists. File hashes and the Node-IDs are of 128-bit length. Index records are stored at nodes with Node-IDs close to the file hash, where closeness is defined by XOR metric. Thus, an index record with file hash FFFFFFFFFFFF has more chances of being stored at a node with Node-ID FFFFFFFFFFFE than a node with Node-ID ABCDEF123456. Overnet uses UDP messages and iterative searches. To locate the closest node to a key, the querying client will send UDP messages to a sequence of node, with each node in the sequence having a Node-ID that is closer to the key.

A. Node Join

To join the network, a peer attempts to connect with at least one peer from a locally cached list of peers from previous sessions. On connecting to such a peer, the joining peer finds its neighbors by sending search messages for its own Node-ID. These peers constitute the routing table of the joining peer. On constructing its routing table from the received replies, it announces its presence to all the peers in its routing table so as to include itself in their routing table. On receiving such an announcement, each peer updates its routing table by including the joining peer in the table.

B. Advertising Files

When a peer joins Overnet, it advertises information about the files it is sharing. In Overnet, the advertisements are published in two steps:

1. In the first step, the peer hashes the content of each file to obtain the file hash. It then sends into the DHT, a location publish message (LPM), which specifies the file hash and the file location. This message is iteratively routed to the peers that are close to the file hash. On receiving these messages, these peers update their local indexes.
$$LPM = \{\ FileHash,\ NodeID,\ IP : Port\ \}$$
2. In the second step, the joining peer hashes each keyword corresponding to the file. The keywords include metadata information such as title, artist, album, file type, file size etc. For each such keyword, the peer sends into DHT a metadata publish message (MPM), which specifies the keyword hash, file hash, and the metadata information about the file. This message is also iteratively routed to the peers that are close to the file hash. On receiving these message, these peers update their local indexes.
$$MPM = \{\ KeywordHash,\ FileHash,\ Metadata\ \}$$

C. Search

To search for a file, the P2P client hashes each keyword provided by the user. The peer then sends a query for each keyword hash, which is iteratively routed through the DHT

to the peers which are close to the keyword hash. When the query reaches a peer that has the record for the keyword, it replies with metadata publish messages for all the files having the keyword. The querying peer receives several results for each keyword. The P2P client filters the results, keeping only those which have all the keywords, and displays the filtered responses to the user. The user selects one of the results for downloading. The peer then searches for the corresponding file hash. The query is again routed iteratively to the peers with Node-IDs closer to the file hash. When the query reaches a peer that has record for the file, it replies with the corresponding location publish message. The querying peer now knows the location of the peer having the file. It then establishes a TCP connection to the location in order to download the file.

1.2 Index Poisoning in Overnet

In index poisoning attack, the aim of the attacker is to make several peers believe that some popular file is present with the victim. To achieve this, the attacker A sends a location publish message to every crawled peer. In these messages, the attacker includes victim's IP address and port number. The attacker puts the file hash of a popular file, which is expected to receive lots of search queries. When a peer B receives such a publish message, it adds this file hash into its index along with the location of the victim. B does not verify whether the victim has the corresponding file or even that A or victim is a participant in the P2P network. When some peer C searches for that file, it may be told by some poisoned peer that victim has the file. C then creates a TCP connection to the victim in order to download the file. The downloading peer then sends a protocol specific message, specifying the file that it wishes to download. Not understanding the message, the victim may ignore it, reply with some error message or may even terminate the connection. Unable to download the file, the downloading peer may retry after some time. Since there will be many peers searching for that popular file, victim will receive a large number of incoming connections, filling up its TCB queue, and therefore making it deny connections to its legitimate users. Moreover, index poisoning attack is also dangerous because of its residue effects: the victim remains under attack even hours after the attacker has stopped poisoning the indexes. This is because the fake records persist in the indexes for hours, even after peers fails to download from the target host.

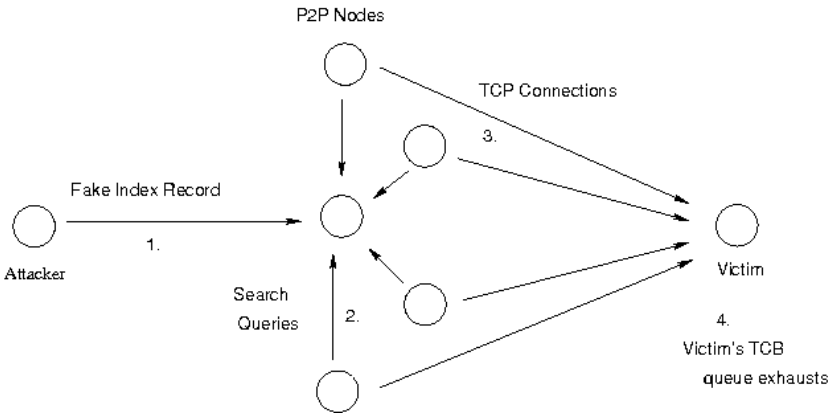

Figure 1: Index poisoning leading to TCP connection DDoS attack

In classic TCP SYN flooding attack, the victim receives incoming SYN requests from spoofed IP addresses. The TCP handshake is never completed since the source IP address is spoofed. This fills up the TCB queue, leading to denial of service to the legitimate users. Such an attack can be prevented by using SYN cookies [15]. However, the DDOS attack described above cannot be prevented using SYN cookies. This is because the source IP address is not spoofed and the three step TCP handshake is always completed.

1.3 Routing Table Poisoning in Overnet

In routing table poisoning attack, the aim of the attacker is to make the peers add victim as their neighbor. To achieve this, the attacker A sends node announcement message to every crawled peer. In these messages, the attacker includes victim's IP address and port number. The attacker puts the node ID such that this ID is close to that of the crawled peer B, so that B includes victim in its routing table. When the peers need to forward any search query or overlay maintenance message, they may forward these messages to the victim. In such a case, the victim receives a flood of messages from the DHT. Moreover, since the victim is not a participant in the P2P network, it will typically respond with an ICMP port unreachable message. Therefore, flooding occurs in both upstream as well as downstream direction. The routing table poisoning generates a burst of messages directed at the victim. After the victim fails to respond, it may be removed from the poisoned routing table.

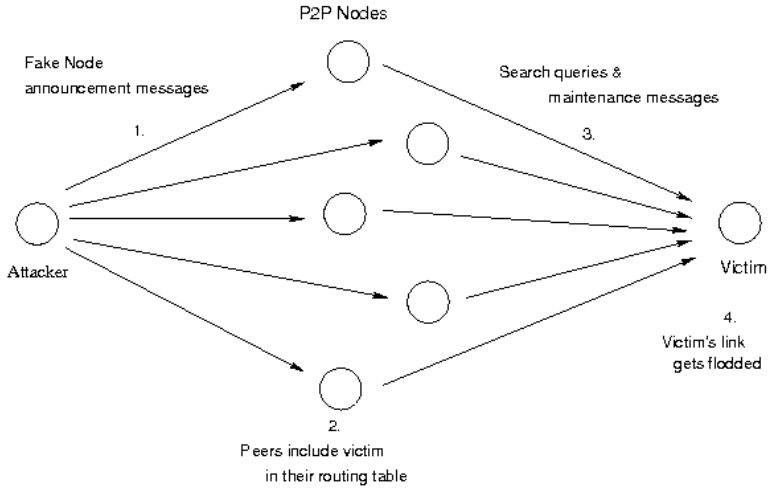

Figure 2: Routing table poisoning leading to link flooding

In routing table poisoning attack, the poisoned peers act as reflectors. A measure of efficiency of a reflection attack [9] is its amplification, which can be defined as the number of packets in the burst. The amplification can be huge in case of routing table poisoning attack. In section 3, we describe a simple yet effective solution to the routing table poisoning attack, that can prevent the poisoning of the routing table and hence the victim's link flooding. Our solution can authenticate a participant of the P2P network with just two messages, one in each direction, without using PKI.

2 Related Work

P2P systems are more vulnerable to a number of well-known threats, such as identity theft (spoofing), violation of privacy, and the man-in-the-middle attack. Threats like index poisoning, routing table poisoning, sybil attacks, etc. are difficult to counter in an environment where membership is dynamic and the presence of central directories cannot be assumed. General P2P security architectures almost exclusively use public key cryptography. These systems provide authentication without anonymity. Research in P2P network security has gained attention only since the past few years. We first discuss some of the exisiting research in P2P security. Then we describe the Identity Based Cryptography (IBC) based approach proposed by Lou and Hwang [6] to defend against index poisoning attack.

Christin et al [4] and Liang et al [5] discuss the pollution attacks in P2P file sharing networks. In a pollution attack, an attacker tampers with the copyrighted content, rendering the content unusable. It then deposits this tampered content in large volumes in the P2P network. In some time, the polluted content gets distributed over a large part of the network. Here, the aim of the attacker is to trick users into downloading polluted copies, users may then become frustrated and abandon P2P file sharing. Pollution attacks are one of the ways to discourage distribution of copyrighted content over P2P file sharing networks. Maniatis et al [10] discuss attrition attacks in P2P networks. An attrition attack is one which tries to reduce the efficiency of the entire system by consuming resources at peers. Douceur [8] discusses sybil attacks in P2P networks. A Sybil attack is one in which an attacker subverts the reputation system of a peer-to-peer network by creating a large number of pseudonymous entities, using them to gain a disproportionately large influence. A reputation system's vulnerability to a Sybil attack depends on how cheaply identities can be generated. The degree to which the reputation system accepts inputs from entities that do not have a chain of trust linking them to a trusted entity, and whether the reputation system treats all entities identically. Daswani and Garcia-Molina [11] study the query-flood DDoS attacks in Gnutella, and the various policies that peers might apply to mitigate the attack. Sit and Morris [16] discuss several routing, storage and retrieval attacks in P2P networks.

Tamassia and Triandopoulos [12] present a new model for content authentication in P2P networks using distributed Merkle trees. The Merkle tree is a widely-used scheme in security applications and cryptographic constructions. The idea is to use a tree and a cryptographic collision-resistant hash function to produce a short cryptographic description of a large data set. Elements of the set are stored at the leaves of the tree and internal nodes store the result of applying a cryptographic hash function to the values of the children nodes. The authentication of an element is performed using a verification path, which consists of the sibling nodes of the nodes on the path from the leaf associated with the element to the root of the tree. The root value is signed and the collision-resistant property of the hash function is used to propagate authentication from the root to the leaves.

Gokhale and Dasgupta [13] present a cryptographically secure representation of trust in P2P networks using RSA accumulators and zero knowledge protocol of modular exponentiation. They describe a cryptographically secure, fault-tolerant, scalable trust model for resilient P2P networks, known as "Troups". In this model, trust is represented by a group membership. This group or "Troup" is formed by creation of RSA accumulator in a distributed manner. Each to-be member generates an exponent, y_i - that is to be accumulated. They, also agree on the seed of the accumulator x, and modulus N. Then, accumulation is done as:

$$z = x^{y_1 * y_2 * \cdots * y_n} \ mod \ N$$

An auxillary value is generated as:

$$aux_i = x^{y_1 * y_2 * \cdots y_{i-1} * y_{i+1} * \cdots * y_n} \ mod \ N$$

The accumulator is public, and is used as troup indentifier. The accumulation serves as a public key of the troup. To verify the membership of troup (z), the member can present the values (aux_i, y_i) to the verifier. The membership is verified if $z = aux_i^{y_i} \ mod \ N$. The knowledge of values (aux_i, y_i) is proved using zero-knowledge protocol for modular exponentiation, such that z can be constructed an verified. This protocol works fine for authenticating peers and verifying their identity, but it cannot restrict a peer from publishing bogus information about files or other peers. Therefore, it cannot thwart index poisoning and routing table poisoning attacks. An attacker can simply join a troup, and begin injecting false information in the P2P network.

Wierzbicki et al [14] present a protocol for authentication in P2P systems using Merkle's puzzles and zero knowledge proofs. Their authentication protocol works in three phases. In the initial phase, a superpeer or a bootstrap creates necessary values for authentication. In the second phase, the file is stored with additional zero-knowledge values that will enable only the owner to modify the file. In the third phase, the owner uses a zero-knowledge proof and Merkle's puzzles to authenticate itself and to safely store the modified file. However, their protocol assumes the presence of a superpeer to assign public and private keys and the identities. Moreover, their protocol does not address the problem where a malicious peer can issue messages on behalf of a non-member.

Lou and Hwang [6] address the problem of index poisoning in P2P file sharing networks using Identity Based Cryptography (IBC). The idea here is to use the Node-ID as the public key to reduce the workload of the certification authority (CA) and to ease the distribution of public key. They describe a notion of peer accountability. A peer is said to be accountable if there exists a way to detect whether an index record has been published by the peer having the file. In other words, for a publishing peer P_b and an index record $D = (F, P_u)$, where F is the file hash and P_u is the peer having the file, a peer is said to be accountable if there exists a binary function $\mathcal{F}$ such that one can detect the following condition:

$$\mathcal{F}(D) = \begin{cases} True, & \text{if } P_u = P_b \\ False, & \text{otherwise} \end{cases} \tag{1}$$

They describe the Reliable Index Exchange Protocol (RIEP) which uses Identity Based Signatures (IBS) to enforce peer accountability. The table below summarizes the notations used in the RIEP:

Table 1: Notations used by RIEP

Notation	Meaning	Notation	Meaning
F	File Hash	G	CA identifier
D	Traditional file index (no RIEP)	R	RIEP enabled file index
P_b	Node-ID of publisher	P_u	Node-ID of an unverified peer
K_p	Private key of peer	K_g	Private key of CA
S_p	Sign using key K_p	S_g	Sign using key K_g
IP_u	IP address of an unverified peer	$Port_u$	Port of unverified peer
T	Timestamp	N	Nonce
E_K	Encryption using key K		

2.1 RIEP

RIEP replaces the currently used index format with Security-Enhanced File Index Format which is defined as follows:

$$R = \{\ D(F, P_b),\ S_b(F, P_b),\ G,\ S_g(P_b)\ \} \tag{2}$$

We now describe the RIEP.

A. Private Key request and assignment

The private key is assigned by the CA to the publisher using symmetric key cryptography. To request a private key, a publisher P introduces a randomly generated nonce as a secret key. The request is then encrypted using the CA's public key. The CA constructs the public key P_b and signs its own private key K_g. The CA then generates the private key K_p for the peer. The public/private key pair and the signature are then encrypted using the nonce and sent to the publisher. This can be explained as follows:

$$P \longrightarrow CA \ :\ \{\ E_g(P_b,\ N)\ \} \tag{3}$$

$$P \longleftarrow CA \ :\ \{\ E_N(P_b,\ K_b,\ S_g(P_b))\ \} \tag{4}$$

B. Index Publishing

An index record is published according to the Security-Enhanced File Index Format. The published record is in the form:

$$R = \{\ D(F, P_b),\ S_b(F, P_b),\ G,\ S_g(P_b)\ \} \tag{5}$$

C. Index Query

When a peer X makes queries to another peer Y for a file index, it signs its queries using its private key K_X. As a reply, peer Y sends back the file index record R along with signed R. This can be explained as:

$$X \longrightarrow Y \;\; : \;\; Q = \{\; X,\; Query(F,T),\; S_X(X, Query(F,T)) \;\} \tag{6}$$

$$X \longleftarrow Y \;\; : \;\; \{\; R,\; S_Y(R) \;\} \tag{7}$$

D. Query Forward

To forward a query to another peer Z, peer Y signs the entire query using its own private key K_Y and sends it along with the query.

$$Y \longrightarrow Z \;\; : \;\; \{\; Q,\; S_Y(Q) \;\} \tag{8}$$

E. Index Verification

Upon receiving an index record from any peer, a peer verifies the signature with a publisher identifier P_b included in the index record. The verifying function $\mathcal{F}$ is defined over RIEP-enabled index record R. Since K_b is generated using the publisher identifier P_b, the following condition is equivalent to the condition $P_b = P_u$ given in Eq. (1).

$$\mathcal{F}(D) = \begin{cases} True, & \text{if } K_b \text{ is used to sign } S_p(D) \\ False, & \text{otherwise} \end{cases} \tag{9}$$

Although RIEP was proposed to defend against index poisoning attacks, it can even defend against routing table poisoning attacks. On joining the P2P network, a peer announces its presence to its neighbors using a node announcement message.

$$NAM = \{\; P_u,\; IP_u : Port_u \;\}$$

If a signed version of the announcement is included with the announcement, the receiver of the announcement can verify it. This can be explained as:

$$NAM' = \{\; P_u,\; IP_u : Port_u,\; S_u(P_u, IP_u : Port_u),\; G,\; S_g(P_u) \;\} \tag{10}$$

The notion of peer accountability introduced by RIEP guarantees that each file index can be traced back to the publishing host without user authentication. Under accountability, a malicious peer can only launch attack against itself or those IP addresses authorized in the past. This greatly limits the attacker's ability of choosing an external target. The disadvantage of RIEP is the extra computational overhead that it incurs on each peer and the need for certification authorities to issue public/private key pairs.

3 Proposed solution for Routing Table Poisoning

We here assume that every peer B in the P2P network maintains some secret value $\mathbb{S}_B$. We use the same notations as used above in RIEP. Whenever any peer B receives a node announcement message from a peer A specifying the existence of a peer V at some location, B can send a message to V transferring the state to V. The insertion of V into the routing table is delayed till V replies back with a valid reply. If V replies, B can verify the authenticity of the announcement. Moreover, B does not need to store the node announcement message that it received from A. It can be reconstructed and verified from V's reply, if received. The entire protocol can be described as

$$A \longrightarrow B \quad : \quad NAM = \{\ P_u,\ IP_u : Port_u\ \} \tag{11}$$

$$B \longrightarrow V \quad : \quad \{\ T,\ f\,(NAM,\ T,\ \mathbb{S}_B)\ \} \tag{12}$$

$$V \longrightarrow B \quad : \quad \{\ NAM,\ T,\ f\,(NAM,\ T,\ \mathbb{S}_B)\ \} \tag{13}$$

Here, f is a one-way hash function. If B receives a reply from V, it can compute the hash of the node announcement mesage NAM, the timestamp T and its secret $\mathbb{S}_B$ and compare it with the hash in the reply. If it matches, V is authenticated. Since, it is computationally hard to compute hash without the knowledge of the secret $\mathbb{S}_B$, on receiving the reply, B can be certain that V is indeed present at the location specified in the node announcement message.

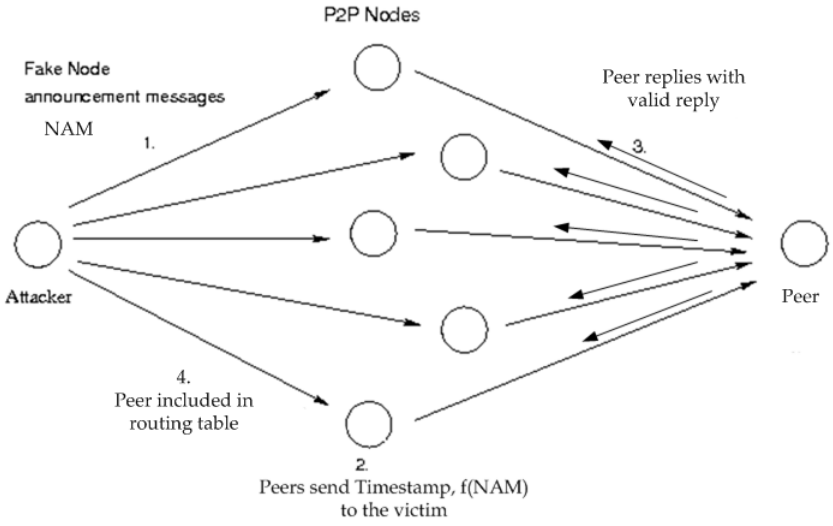

Figure 3: Peer replies with valid reply and is included in routing table

Now, consider the case when some attacker A sends a fake node announcement message to several peers $B_1, B_2, B_3, \ldots, B_n$. In such a case, each peer sends a message to V in order to check its authenticity. Since V is not a participant in the P2P network, either it will reply with an error message or it will not reply, and hence will not be authenticated. This prevents the poisoning of the routing table. Moreover, in such a case, for every message that will be sent to V, A will have to send a message to some peer B_i. Earlier,

an attacker would poison routing tables of peers one at a time, but here, an attacker trying to flood the victim's link will have to send fake node announcement messages at the same time, which would flood its own link. The P2P network in such a case only acts as a reflector, but not as an amplifier. Hence, the attack would fail.

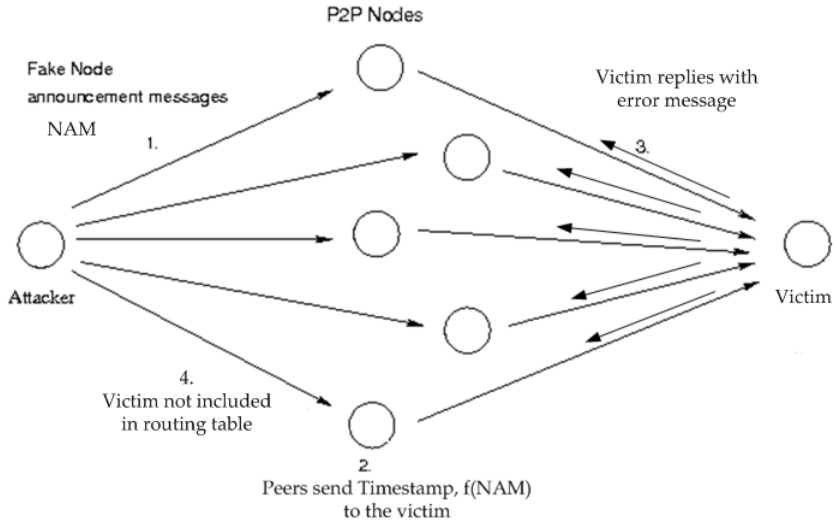

Figure 4: Victim replies with error message and is not included in routing table

The advantage of the above solution is its simplicity, and ease of implementation. Moreover, the peers are authenticated without using PKI, which has a huge computational overhead. The only drawback here is that if V is not a participant in the P2P network, some part of the traffic is still redirected to it.

4 RIEP Implementation

The changes brought by RIEP is applicable to all P2P file sharing networks, despite their differences in design. We here discuss some RIEP implemenation issues:

4.1 RIEP Implementation Requirements

RIEP can deployed incrmentally in a P2P file sharing network. To maintain backward compatibility with existing P2P file sharing networks, RIEP sends all its messages in plaintext. The only changes made to the file index are signatures. For non-RIEP peers, the signatures in the file index as well as in the exchanged messages can be ignored, while RIEP-enabled peers gain the capability of defending against index poisoning attacks.

4.2 Private versus Public IP addresses

RIEP uses a public IP address as a part of its public key of peer. Most P2P file-sharing users are home users. They connect to Internet using broadband connection from their

ISP. In most cases, these users use a firewall/router to establish home networks, and the P2P hosts are clients of the routers. Therefore, a P2P host's own IP address cannot be used to establish connection. A certification authority (CA) can a identify the public address of a peer from its private key request packet. Since the same address is used to establish connection with other peers, RIEP uses this address as a part of the public key. Using public IP address can help defend against a list of attacks. Using the entire endpoint as opposed to only the IP address can help distinguishing peers when multiple peers are behind the same firewall, as many home networks have multiple computers that access the Internet simultaneously.

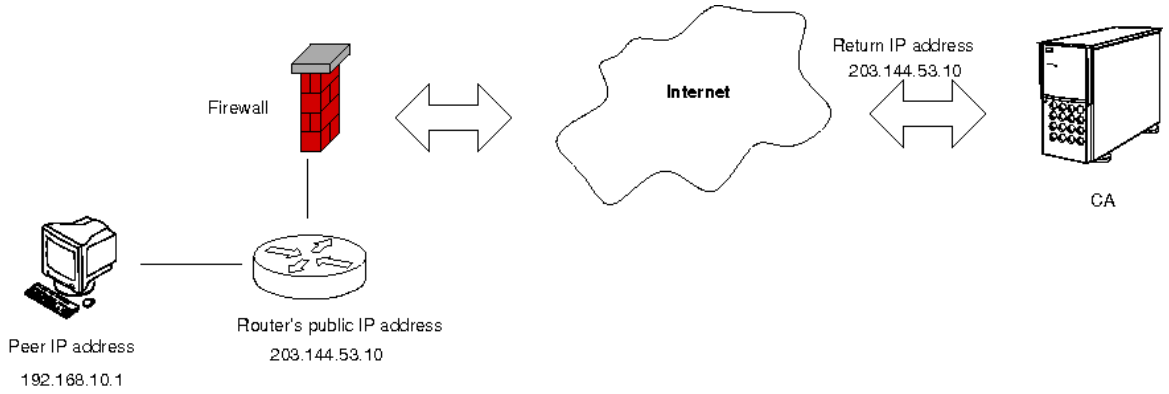

Figure 5: The public IP address seen by the CA is used as a part of the public key

4.3 Multiple Certification Authorities

RIEP supports multiple certification authorities. The CA's identifier is used to identify individual CA's. When requesting private key, a peer can choose from the list of well known, trusted CA's. A chosen CA will sign the returning message with its private key. This signature is also a part of the file index R. A peer receiving the file index R can then use a verification fucntion corresponding to CA's identifier to verify peer's accoountability.

Any single CA has the capability to sign all the RIEP messages. If an attacker can masquerade itself as a valid CA, it will bring down the entire peer accountability structure established by RIEP. Because of this threat, the identities of all CA's must be further protected. Each CA must register its identity and its public key with a trusted certificate authority, and all peers will verify CA's identity with the certificate authority.

4.4 Implementation Complications and Limitations

RIEP provides effective defense against index poisoning DDoS attacks. However, there are complications caused by changing IP addresses, peer behind multiple routers, and private key revocation, that are inherent to P2P file sharing networks. These problems may impose some limitations to apply the RIEP.

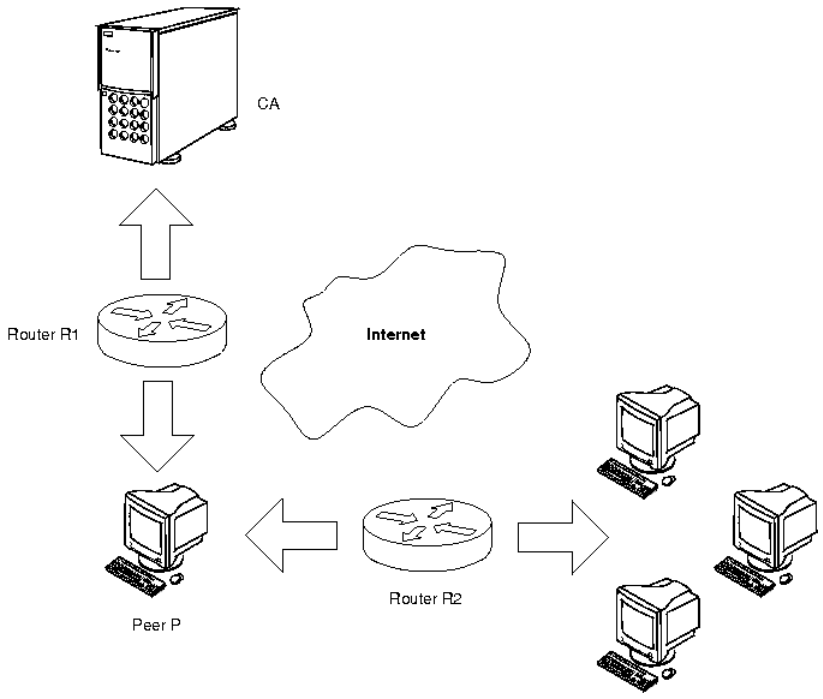

Figure 6: A peer behind multiple routers will have problem publishing file index under RIEP

In many organizations, there are multiple routers connecting the Intranet to the Internet. In such cases, RIEP may stop a peer from exchanging index with other peers that are behind other routers in the same organization. This is because the private key will be assigned to a peer P based on its public IP address. But when it wishes to exchange a file index with other peers that are in the same organization, the traffic to those peers may be forwarded from within the Intranet, due to the presence of routers within the Intranet. Since the signatures will match, the receiving peers accept the file indexes. However, when they try to contact peer P via claimed endpoint, their requests are sent to router $R1$ instead of router $R2$, since the claimed IP address belongs to router $R1$. The peer P accepts incoming connection via a forwarded port of router $R2$, while it does not have any open ports in router $R1$. This causes other peers to accept the index, but fail to establish connections.

In most cases, such a network configuration occurs in a corporate environment, where P2P file sharing is discouraged, if not completely blocked, due to security and legal concerns. Furhermore, most popular P2P file sharing networks in use today face the same problem. With RIEP, a peer behind multiple routers can still function well as a leecher, except it cannot provide content to other peers.

5 Conclusion and Future Work

We discussed how an attacker can exploit P2P index and routing substrates for DDoS attacks. It is difficult for a P2P file sharing network to authenticate advertisements, that

is, to verify that the advertised content or peer is indeed present at a location. At the time of writing this report, the two most widely being used P2P file sharing netwoks - Overnet and FastTrack were found susceptible to the index poisoning and routing table poisoning [1] [2]. Overnet is a core component of eDonkey2000 and eMule. FastTrack is being used by several popular clients such as Kazaa, Kazaa-Lite, Grokster, and iMesh, having more than three million active users.

P2P file sharing has a broad range of business and educational applications. Without special care, the advertisements in these systems can easily be compromised. When designing P2P file sharing systems in the future, it is crucial that designers bear poisoning attacks in mind, since they also hamper the efficiency of the entire network by injecting false information. We believe that countermeasures based on advertisement authentication will be largely successful.

Bibliography

[1] N. Naoumov, and K.W. Ross, Exploiting P2P Systems for DDoS Attacks, International Workshop on Peer-to-Peer Information Management, Hong Kong, May 2006

[2] J. Liang, N. Naoumov, and K.W. Ross, The Index Poisoning Attack in P2P File-Sharing Systems, Infocom 2006, Barcelona, 2006

[3] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. IPTPS 2002.

[4] N. Christin, A. S. Weigend, and J. Chuang. Content Availability, Pollution and Poisoning in Peer-to-Peer File Sharing Networks. Proceedings of the Sixth ACM Conference on Electronic Commerce (EC05) Vancouver, BC, Canada. June 2005.

[5] J. Liang, R. Kumar, Y. Xi, K.W. Ross. Pollution in P2P File Sharing Systems, IEEE INFOCOM 2005, Miami, March 2005

[6] Xiaosong Lou and Kai Hwang, "Prevention of Index-Poisoning DDoS Attacks in Peer-to-Peer File-Sharing Networks," submitted to IEEE Trans. on Multimedia, Special Issue on Content Storage and Delivery in P2P Networks, November, 2006.

[7] D. Dumitriu, E. Knightly, A. Kuzmanovic, I. Stoica, and W. Zwaenepoel. Denial-of-Service Resilience in Peer-to-Peer File-Sharing Systems, Proceedings of ACM SIGMETRICS 2005, Banff, Canada, June 2005.

[8] J. Douceur. The Sybil Attack. In Proceedings of the 1st International Workshop on Peer-to-Peer Systems, pages 25160, Boston, MA, USA, Mar. 2002.

[9] V. Paxton. An Analysis for Using Reflectors in Distributed Denial-of-Service Attacks. ACM SIGCOMM Computer Communication Review, July 2001, pp.38-47.

[10] Petros Maniatis, TJ Giuli, Mema Roussopoulos, David S. H. Rosenthal, and Mary Baker. Impeding Attrition Attacks in P2P Systems. In Proceedings of the 11th ACM SIGOPS European Workshop, Leuven, Belgium, September 2004. ACM SIGOPS.

[11] N. Daswani and H. Garcia-Molina. Query-Flood DoS Attacks in Gnutella. CCS 2002.

[12] R. Tamassia and N. Triandopoulos, Efficient Content Authentication over Distributed Hash Tables, Technical Report, Brown University, November 2005.

[13] S. Gokhale and P. Dasgupta, "Distributed Authentication for Peer-to-Peer Networks," in Proceedings of IEEE Workshop on Security and Assurance in Ad hoc Networks, 2003.

[14] A. Wierzbicki, A. Zwierko, and Z. Kotulski. Authentication with controlled anonymity in P2P systems. In Proceedings of the Sixth international Conference on Parallel and Distributed Computing Applications and Technologies (December 05 - 08, 2005). PDCAT. IEEE Computer Society, Washington, DC, 871-875.

[15] W. Eddy. TCP SYN Flooding Attacks and Common Mitigations. (Internet Draft) http://tools.ietf.org/html/draft-ietf-tcpm-syn-flood-00

[16] Emil Sit and Robert Morris. Security Considerations for Peer-to-Peer Distributed Hash Tables. In IPTPS 2002.