Prevention of Index-Poisoning DDoS Attacks in Peer-to-Peer File-Sharing Networks*

Xiaosong Lou, *Student Member IEEE* and **Kai Hwang,** *Fellow IEEE*

University of Southern California, {xlou, kaihwang}@usc.edu

Abstract: A major security threat to the normal use and legal sharing of *peer-to-peer* (P2P) resources is DDoS (*distributed denial-of-service*) attacks caused by file index poisoning. This type of attacks exploits the design vulnerability of P2P networks. By populating unprotected peers with poisoned file indexes, attacker can cause DDoS flooding attacks on arbitrary hosts, even outside of the P2P network. We solve the index-poisoning problem in P2P file-sharing systems using identity-based cryptography to establish peer accountability. With accountability, all peers remain anonymous but their file indexes can be traced back to the original sending address without contacting the sender.

We prove that accountability effectively prevents a peer from launching index-poisoning attacks. A new *reliable index-exchange protocol* (RIEP) is proposed to enforce peer accountability. This protocol is applicable to all P2P file-sharing networks, either structured or unstructured. The system is designed to allow gradual transition of peers to become RIEP-enabled. We develop analytical models to characterize the poison propagation patterns with and without RIEP defense. These poisoning models are validated by simulated RIEP experiments on P2P networks over one million nodes. The reported experimental results support the advantages predicted by the theoretical models. These results prove the effectiveness of using RIEP to prevent index-poisoning DDoS attacks. We also discuss the implementation of the RIEP scheme and its limitations in real-life P2P networks.

Keywords: ***Peer-to-peer networks, distributed file sharing, content delivery, index poisoning, DDoS attack, identity-based signatures, network security, and public key cryptography***

---

* Manuscript submitted to *IEEE Transactions on Multimedia,* special issue on Content Storage and Delivery in P2P Networks, *Nov.9*, 2006. This work was supported by NSF Grant ITR-0325409 at the University of Southern California. Corresponding author is Kai Hwang at Email: kaihwang@usc.edu, Tel. 213 740 4470 and Fax: 213 740 4418.

1 Introduction

In P2P file-sharing networks, client software maintains table of file indexes that maps file identifier to sharing hosts. With index poisoning, the attacker publishes false (poisoned) file indexes to unprotected peers. The file download requests from poisoned peers cause massive amount of misguided network traffic to the victim host. Naoumov et al [22] discovered that with a large number of poisoned file indexes, an attacker can launch DDoS attacks against arbitrary host in the Internet.

P2P file-sharing networks has become the most popular Internet content delivery systems [23]. As an example, more and more open source software choose P2P networks such as BitTorrent and eMule as the publishing platform. A study conducted in October 2005 [20] found that the number of P2P file-sharing software installations has exceeded 6 millions in US and 9 millions worldwide. This population of P2P networks makes it an important topic for Internet security research [26].

P2P researchers have long identified the use of common client software as a major security threat to the network. The attacker can exploit imperfection in P2P client software and launch attacks. More and more attacks that target P2P file-sharing networks are surfaced each year. Furthermore, the distinct features of P2P networks present some unique security vulnerabilities. For example, P2P client software usually caches IP addresses of recently accessed peers. Once a vulnerability is discovered in the common client software, the attacking program is much easier to propagate since most likely all the peers in the cache have exactly the same vulnerability [28].

Among these attacks, index poisoning DDoS attacks [17] are particularly threatening. Figure 1 shows such flooding attacks in a P2P file-sharing environment. The attacker first spreads poisoned file indexes to infect many peers in the P2P network. The infected peers will act as Zombies when they send file download requests towards the victim host. These massive requests generate a large number of packets forming a DDoS flood to attack the victim. The victim host could be located either inside or outside the P2P system.

In most P2P files-sharing networks, the peers are anonymous to the publisher. They only expose their endpoints (*IP addresses and port numbers*) to peers to establish connection. While a user ID is part of the communication, this ID is based on assertion without any verifiable authentication. The delivery is accomplished even when the publisher has long left

the network, as long as there are sufficient peers left to share the file. This anonymity feature is critical to the success of P2P file-sharing networks.

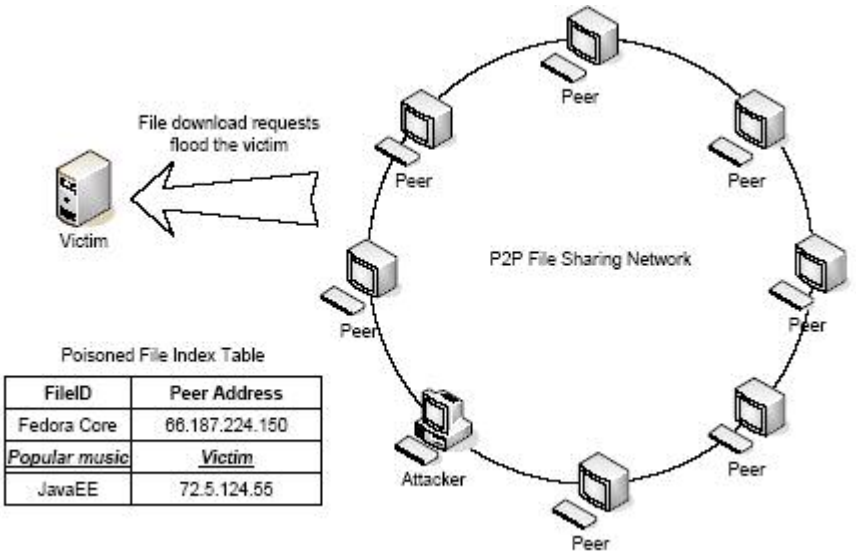


Figure 1 Scenario of index-poisoning DDoS attacks: Attacker spreads poisoned file indexes in the P2P network, causing massive file download requests targeted at the victim located either inside or outside the P2P file-sharing system

Currently, popular P2P networks include BitTorrent, eMule, Gnutella, Fasttrack, and Freenet. There are two network organizations among these P2P file-sharing networks: *pseudo-distributed* systems like BitTorrent and *distributed* systems such as Chord. In BitTorrent, a tracker is present to maintain file index information and distribute file index entries upon peer request. In unstructured networks and in DHT overlays, there is no tracker. Each peer maintains its own file index table.

In this paper, we introduce a *Reliable Index Exchange Protocol* (RIEP) into P2P file-sharing networks. Based on *identity-based signature* (IBS), RIEP guarantees *peer accountability* of index publishing and exchange, i.e. any file index can verifiably be traced back to its original publisher's address without contacting the original host. We demonstrate that RIEP successfully protects P2P network from index poisoning attacks, while maintains the same level of anonymity in current P2P file-sharing networks. Notations and abbreviations used in the paper are summarized in Table 1.

Table 1 Notations and Abbreviations used in Paper

Notation	Meaning	Notation or Abbreviation	Meaning or Definition
F	File identifier (File ID)	r	Peer infection rate (infected peers/sec)
G	Identifier of PKG	T	Attack window size, $T = n/r$ (sec)
D	Traditional file index (no RIEP)	$\lambda(t)$	Poison detection rate at time t
R	RIEP-enabled file index	E_K	Encryption of a message with key K
P_a, P_b, P_u	Identifiers for attacker, publisher, & an unverified peer, respectively	p	Probability of an infected peer attempting file download using poisoned index
K_p, K_a, K_g	Private keys for a peer, attacker, and PKG, respectively	S_a, S_p, S_g	Digital signatures using keys K_a, K_p, and K_g, respectively
$I(t)$	No. of infected peers up to time t	PKG, CA	Private key generator, certificate authority
α	RIEP enabling ratio (percentage)	IBS, IBC	Identity-based signature or cryptography
n	P2P network size (peer count)	RIEP	Reliable index exchange protocol
β	Peer poisoning ratio, $\beta = I(t)/n$	DDoS	Distributed denial-of-service attack
θ	Attack traffic rate (Mbps)	PKI	Public key infrastructure

This paper is organized as follows: Section 2 reviews the related works in P2P security, index poisoning, and identity-based cryptography. Section 3 presents the new RIEP protocol to solve the index poisoning problem. Security-enhanced index format, peer messaging protocol specification, and index validation procedure are given. Section 4 provides theoretical models of the index- poisoning effects. We also provide a security analysis of the RIEP. Section 5 discusses implementation requirements and the protocol limitations. Section 6 evaluates performance of the RIEP through simulation experiments. Section 7 concludes with a summary of research contributions and comments on further research needed.

2 Related Work

P2P applications has attracted interests from both industry and research community. Gummadi et al [14] discovered in KaZaA that P2P traffic deviates substantially from power law distribution. Manini et al [19] summarizes a list of proposals to model P2P file-sharing systems. In industry, CachLogic[5] has conducted extensive traffic analysis on different P2P applications. Facetime [12] monitors new security threats affecting P2P communications.

Until recently, users have not realized the unique security concerns facing P2P file-sharing networks. In a government hearing in 2003, Davidson [10] claims that P2P software has exactly the same security risks as any other software. Many researchers disagree with this

rather naïve assessment. For example, Wallach [26] presents a list of security issues that are unique in P2P routing protocols.

Douceur [11] points out that *sybil* attacks are very difficult to avoid without a centralized authority. Using a graph formulation, Cheng et al [8] show that symmetric sybil-proof reputation function does not exist. Zhou et al [28] suggest that P2P worm can be much more efficient than random-scan worms because of local cache of peer addresses. Chen et al [7] propose a simulation framework to better understand the propagation of non-scanning worms. Maniatis et al [18] use the term *attrition attacks* to describe DDoS attacks.

Structured P2P networks organize peer nodes by means of *Distributed Hash Table* (DHT). This DHT overlay present even more security challenges on top of unstructured P2P networks. Srivatsa et al [25] presented a list of these DHT attacks, and provided quantitative analysis of the potential damages. Castro et al [21] discussed routing challenges in P2P DHT overlay.

Naoumov et al. proposed a DDoS attack scheme that further exploits P2P file-sharing networks. Because P2P file-sharing software probe each other for file pieces, the attacker can inject fake file index information into the network, and point those fake index entries to the victim host. When peers following these poisoned file index entry and probe the victim, it is flooded with file download requests.

Naoumov's experiments suggest that P2P networks can be exploited to launch DDoS attacks against not only other peers, but also arbitrary host that are not even part of the P2P network. Index poisoning attacks is dangerous also because of its residue effects: the victim remains under attack even long after the attacker has left the P2P network.

In P2P file-sharing networks, all peers are anonymous. This anonymity feature is very important to the popularity of these networks, and is in conflict with normal user authentication approaches. To further enhances peer anonymity of peers, Androutsellis-Theotokis et al [1] presented a list of proposed techniques in their survey, including disassociation of content source and requestor [9], anonymous connection layers [13] and censorship resistant lookup [15].

Since peers are anonymous, identity management is often ignored in P2P networks; therefore the possibility of sybil attack rises. In some systems, key distribution mechanisms are introduced to overcome the lack of authentication, such as in *Oceanstore* [16]. While

these approaches are useful for access control, they do not provide any defense to malicious peer launching index-poisoning attacks.

The key to defend index poisoning is to provide a verifiable identity via cryptography. *Public key infrastructure* (PKI) has been applied by the P2P community[3]. PKI puts a huge amount of workload on the *certificate authority* (CA) in P2P networks, because of large peer population and free peer joining and leave. An useful alternative to PKI is the *identity based cryptography* (IBC) [2]. Because user identifiers are used as public key, IBC greatly reduces the workload of the *private key generator* (PKG) [2, 4, 6, 24].

Due to the characteristics of P2P file-sharing networks, IBC rises as a natural replacement for PKI in P2P networks. Wen et al [27] have introduced IBC in P2P group admission control. One aspect of IBC, the IBS is particularly useful in our work. This signature scheme was originally developed by Shamir [24] prior to the realization of the complete IBC infrastructure. In this paper, we adopt the IBS scheme in Cha et al [6], which is based on bilinear pairing.

3 Reliable Index Exchange Protocol (RIEP)

Peer anonymity and dynamic membership are two salient features of P2P file-sharing network. With these two features, it is very difficult to enforce peer authentication in a P2P network. However, to defend against index poisoning attacks, a level of reliability on the file index entries must be established. We introduce a new concept of *peer accountability* in P2P file-sharing networks to bridge the gap between anonymity and need for reliability.

3.1 Peer Accountability

In P2P file sharing networks, a *file index D* maps the *file identifier F* with the *peer identifier* P_u that is sharing the file. In many existing networks, endpoint address is used to identify a peer. In general, we denote a file index D = (F, P_u), where F is the file identifier and P_u denotes the peer identifier. The peer that publishes D is called an *index publisher* denoted by P_b. Index poisoning attacks exploit the fact that there is no direct reference to P_b in the index D, and P can be modified to point to an attack target. To eliminate this vulnerability, we call a P2P network peer *accountable,* if there exists a binary function $\mathcal{F}$ such that one can detect the following condition:

$$\mathcal{F}[D] = \begin{cases} 1\,(\text{true}), & \text{if } P_u = P_b \\ 0\,(\text{false}), & \text{otherwise} \end{cases} \quad \textbf{(1)}$$

With peer accountability, the file index itself contains enough information to detect the condition $P_u = P_b$. Therefore, the attacker can no longer point a poisoned file index to the victim IP address. Thus the threat of index poisoning attack is relieved. While our discussion assumes unstructured P2P networks, this level of accountability can also help in the secure routing of structured DHT overlay networks. Accountability proves to be a viable option in securing P2P file-sharing operations, where reliability is desired but user authentication is impossible.

We use digital signature to enforce peer accountability. Because of the huge number of peers, each peer can be either the server or the client, and they randomly join and leave the network, the CA workload becomes cost-prohibitive. With identity-based signature, however, the problem is easily solved. Since the file index contains the publisher identifier P_b, we can use this identifier as the public key. The IBS significantly reduces the workload to PKG. In PKI for an unstructured P2P network, the overhead is O(n^2), while in the IBS system the overhead is reduced to O(n), where n is the network size. Because every peer needs to contact the CA only once for private key generation while public key is implicit.

3.2 The RIEP Protocol

In this section, we specify the RIEP protocol formally. This protocol is based on an identity-based signature scheme. In IBS, the identity that generates public/private key pair is called PKG, which is similar in functionality of the CA in PKI. The notation S_P (F, P_b) indicates a signature generated with the use of the publisher private key P_b on the index D = (F, P_b). Similarly, E_g (P) represents the encryption of a peer address P with the key K_g. The RIEP scheme consists of two parts: The extended file index format and a series of messages and the procedure for peer index exchange.

A. Security-Enhanced File Index Format

The *RIEP-enabled file index* R is modified from the traditional file index D = (F, P) by having three extended fields as follows:

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

Table 2 summarizes the field names and notations used. The publisher signature S_b is used to enforce peer accountability, while the PKG identifier G and PKG signature S_g are needed to accommodate multiple PKGs, as well as to provide security for the PKG identity.

Table 2 Security-Enhanced File Index Format

File Identifier	Peer Identifier	Publisher Signature	PKG Identifier	PKG Signature
F_0	P	$S_b(F_0, P_b)$	G_0	$S_g(P_b)$
F_1	P	$S_b(F_0, P_b)$	G_1	$S_g(P_b)$

B. RIEP Protocol Messages and Procedures

As illustrated in Fig.2, the RIEP protocol specifies the file classes and messages formats for *private key request, index publishing, index query and reply* as well as for *index forwarding.* An index verification procedure is then followed. There are five entities involved in this protocol specification in Fig.2. A peer may assume different roles in the exchange of file indexes. The publisher issues file indexes. Receiving peer receives published indexes and stores in local cache. A query peer seeks the index entry among its neighbors. A forwarding peer forwards queries from other peers.

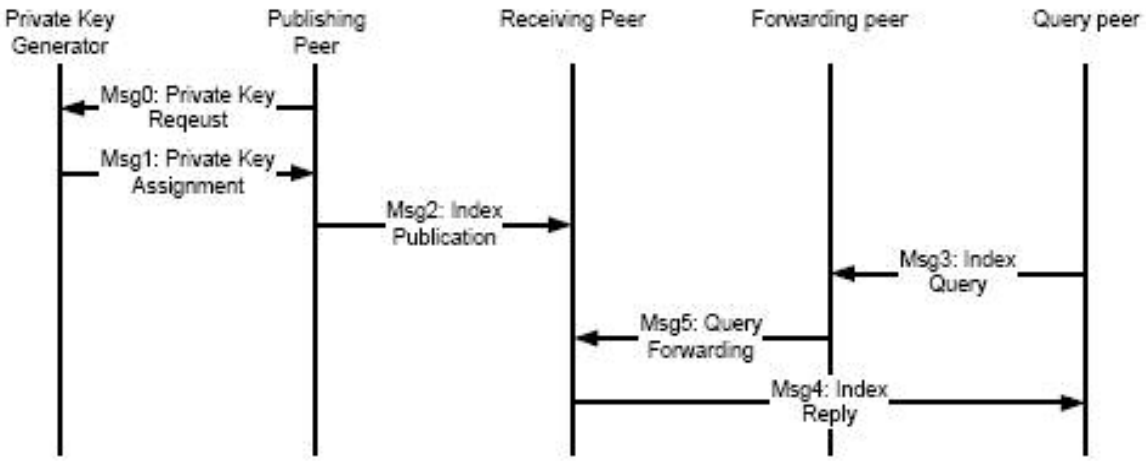


Figure 2 Specification of the RIEP protocol in 6 steps and 6 messages exchanged

(1) Private key request and assignment:

The RIEP assures secure private-key assignment from PKG to publisher via symmetric cryptography. In the request, publisher includes a randomly generated *nonce* as the secrete key. The requested message is then encrypted using PKG's public key in message Msg0. The

PKG constructs the public key from publisher identifier P_b and signs this public key with the private key K_g of PKG. The system then generates the peer private key K_p. The public/private key pair and the signature are then encrypted using the nonce and sent to the publisher in Msg1. These two messages are formally specified below:

$$\text{Msg0} = E_g\,(P_b\,,\, nonce) \qquad \textbf{(3.a)}$$

$$\text{Msg1} = E_{nonce}\,[P_b,\ K_b,\, S_g(P_b)] \qquad \textbf{(3.b)}$$

(2) Index publishing:

After receiving Msg1, the publisher constructs the file indexes according to RIEP index format. The published file index is in the form of

$$\text{Msg2} = R = D(F,\, P_b),\, S_b(D),\, G,\, S_g(P_b) \qquad \textbf{(4)}$$

(3) Index query:

When a peer (*X)* makes queries to another peer (*Y*) for a file index, it signs its query using key K_X (Msg3). Peer Y sends the entire file index entry, including the signature of the publisher, PKG identifier *G*, and PKG signature S_g, as well as digitally signs the entire message using key K_Y (Msg4). These two messages are specified below:

$$\text{Msg3} = X,\, \text{query}(F,\, time),\, S_X\,[\text{X},\, \text{query}(F,\, time)] \qquad \textbf{(5.a)}$$

$$\text{Msg4} = R,\, S_Y(R) \qquad \textbf{(5.b)}$$

(4) Query forward:

Peer *X* sends a query to peer *Y* using Msg3. In some implementation, peer *Y* will forward this query to its neighbor peer *Z*. Similar to index query, the forwarded message must be signed. However, previously forwarded peer signature is no longer needed. In this case, *Y* signs *X*'s query with its own signature before forwarding the query to its neighbors in Msg5. Peer *Z* replies to the query by sending a message Msg4 to *X*.

$$\text{Msg5} = X,\, \text{query}(F,\, time),\, S_X\,[X,\, \text{query}(F,\, time)],$$
$$S_Y\,\{X,\, \text{query}(F,\, time),\, S_X\,[X,\, \text{query}(F,\, time)]\} \qquad \textbf{(6)}$$

(5) Index verification Procedure:

Upon receiving an index entry from any peer, a peer verifies the signature with a publisher identifier P_b included in the file index. The file index entry is validated, if the verifying function $\mathcal{F}$ returns with a true value, otherwise it is considered poisoned. If the file

index entry does not contain any signature, then the publisher is not RIEP-enabled. A verification procedure enforces acceptance criteria of received file indexes.

In an IBS scheme, verifying a digital signature involves solving a *Decisional Diffie-Hellman Problem* (DDHP) [6]. The verifying function $\mathcal{F}$ is defined over a RIEP-enabled file index R. Because K_b is generated using the publisher identifier P_b, the following testing condition is equivalent to the condition $P_b = P_u$ given in Eq.(1).

$$\mathcal{F}[R] = \begin{cases} 1\,(\text{true}), & \text{if } K_b \text{ is used to sign } S_P(D) \\ 0\,(\text{false}), & \text{otherwise} \end{cases} \qquad \textbf{(7)}$$

4 RIEP Performance Model and Security Analysis

In this section, we provide theoretical models of the propagation of poisoned file indexes in both traditional P2P networks and RIEP-enabled P2P networks.

4.1 RIEP Performance Model

In an index-poisoning attack, the attacker sends poisoned file indexes to other peers discovered in a peer's local cache. When a peer is infected, it attempts to download a file using the poisoned index with a probability p. The probability p represents the popularity of the file among all peers and thus $p << 1$. Obviously the download request will fail. Then the peer would regard the file index as stale data and the poisoned index is discarded. To simplify the discussion, we assume the poison contains only one index.

Assume there are n peers in the network and the attacker has a *peer infection rate* of having r peers poisoned per second. The attacker takes $T = n/r$ time units to complete the attack. We first derive the propagation model of poisoned file indexes.

Lemma 1: Let $I(t)$ be the number of infected peers up to time t and p be the attack probability of a given index file. The *detection rate* of poisoned indexes is estimated by:

$$\lambda(t) = pI(t) \qquad \textbf{(8)}$$

Proof: At time t, the total number of infected peers is $I(t)$. Since each peer has the probability p to attempt a file download, the total download attempts equals the product $pI(t)$. Each failed download leads to a peer to detect the poisoning of indexes, thus $\lambda(t) = pI(t)$. ***Q.E.D***

Theorem 1: Let *T* be the attack time window. Given the attack probability *p* and the peer infection rate *r*, the number of infected peers at time t is calculated by:

$$I(t) = \begin{cases} \frac{r}{p}(1\text{-}e^{\text{-}pt}), & 0 < t < T \\ \frac{r}{p}e^{\text{-}pt}(e^{pT}\text{-}1), & t \geq T \end{cases} \quad \textbf{(9)}$$

Proof: Within window $0 < t < T$, we have $\frac{dI(t)}{d(t)} = r - \lambda(t) = r - pI(t)$. Integrating both sides, we have $ln[r\text{-}pI(t)] = -pt + c$. With initial condition $I(0)=0$, we have $I(t) = \frac{r}{p}(1 - e^{-pt})$. After time *T*, we have $\frac{dI(t)}{d(t)} = -\lambda(t) = -pI(t)$. Because *I*(t) must be continuous at time *T*, we have the expression $I(t) = \frac{r}{p}e^{-pt}(e^{PT} - 1)$. Combining the above, we obtain Eq.(9). ***Q.E.D***

Corollary 1: The maximum number of infected peers is estimated by:

$$Max[I(t)] = \frac{r}{p}(1 - e^{-pT}) \quad \textbf{(10)}$$

Proof: When $0 < t < T$, $\frac{dI(t)}{d(t)} = re^{-pt} > 0$; when $t > T$, $\frac{dI(t)}{d(t)} = -re^{-pt}(e^{PT} - 1) < 0$. Therefore, $Max[I(t)] = I(T) = \frac{r}{p}(1 - e^{-pT})$. ***Q.E.D.***

With the protection offered by RIEP protocol the attacker will not be able to infect a RIEP-enabled peer with poisoned indexes. During $0 < t < T$, the attacker sends poisoned index to all peers at a rate $r=n/T$. Assume RIEP-enabled peers are randomly distributed among all peers and there is α percentage of RIEP-enabled peers among all peers in the network. We derive below a poison propagation model for file indexes.

Theorem 2: In an RIEP-enabled P2P network, the accumulative number of infected peers is calculated by:

$$I(t) = \begin{cases} \frac{r(1-\alpha)}{p}(1\text{-}e^{\text{-}pt}), & 0 < t < T \\ \frac{r(1-\alpha)}{p}e^{\text{-}pt}(e^{PT}\text{-}1), & t \geq T \end{cases} \quad \textbf{(11)}$$

Proof: For $0 < t < T$, only $(1\text{-}\alpha)$ of the peers contacted by the attacker can be poisoned, $\frac{dI(t)}{d(t)} = r(1-\alpha) - \lambda(t) = r(1-\alpha) - pI(t)$. For $t > T$, we have $\frac{dI(t)}{d(t)} = -\lambda(t) = -pI(t)$. Solving these two equations, we obtain the expressions in Eq.11. ***Q.E.D.***

Corollary 2: Given the parameters α, r, and p as in Table 1. The maximum number of infected peers in a RIEP-enabled network is equal to

$$Max[I(t)] = \frac{r(1-\alpha)}{p}(1-e^{-pT}) \qquad \textbf{(12)}$$

Proof: For 0 < t < T, $\frac{dI(t)}{d(t)} = r(1-\alpha)e^{-pt} > 0$. For t > T, $\frac{dI(t)}{d(t)} = -r(1-\alpha)e^{-pt}(e^{PT}-1) < 0$.

Therefore, we have $Max[I(t)] = I(T) = \frac{r(1-\alpha)}{p}(1-e^{-pT})$. ***Q.E.D.***

Theorems 1 and 2 prove that the RIEP protocol reduces the speed of propagating poisoned indexes through the P2P network. Specifically, the speed of propagation and the maximum number of infected peers are reduced directly proportional to the RIEP enabling ratio α. In section 6, we will simulate RIEP-enabled network and show that the modeling given in Theorems 1 and 2 are indeed validated rather well by the simulation results.

4.2 Index Poisoning Attack Scenarios

We analyze the security features of RIEP by discussing two attacking scenarios. These are often encountered in real-life P2P systems such as the BitTorrent, KaZaA, etc. We explain below how the RIEP protocol handles each of these attack scenarios, separately.

A. Private Key Self-generating

In IBS, generating the public key/private key pair is based on well-publicized mathematical algorithms. Because endpoint is the public key, there is nothing to prevent an attacker from generating the private key corresponds to any address it plans to attack. It can then sign poisoned file indexes with self-generated private keys and launch attack.

RIEP provides defense against such attack by including $S_g(P_b)$ in the index format. Communication between a peer and PKG is protected by PKI. Since there are relatively a very small number of PKGs, and such communication happen only once per peer session, the overhead is limited. Any peer receiving R can detect if a proper signature by the PKG is present. PKI prevents the attacker from acquiring K_g, thus poisoned index will be detected.

B. Private Key Caching

To relieve the workload of PKG, RIEP encourages private key caching, if the peer has not changed its IP address and port number since the last time it joined the P2P network. As a consequence, an attacker can cache all the private keys it received from PKG after every IP

address change. This will enable the attacker to launch DDoS attacks against one or many of those IP addresses.

A more advanced attacking scheme is called collaborative key caching. In this attack, a number of attackers residing in different networks form a collaborative outside the P2P network. They exchange their cached private keys among each other, and an attacker can then launch attach to victims outside of its local network. Although this type of collaborative attack avoids attacking a local victim, it still cannot select a specific target victim, because of the limited number of cached private keys.

With collaborative key cashing, the attacker can launch index-poisoning attacks against a remote network, provided that at least one of the collaborators caches one of the private keys within the target network. We limit attacker's caching capability by introducing a timestamp in each peer identifier. This timestamp is provided by PKG when the private key is requested. With this timestamp and a key expiration scheme, we limit the target to the most recently cached attacker addresses.

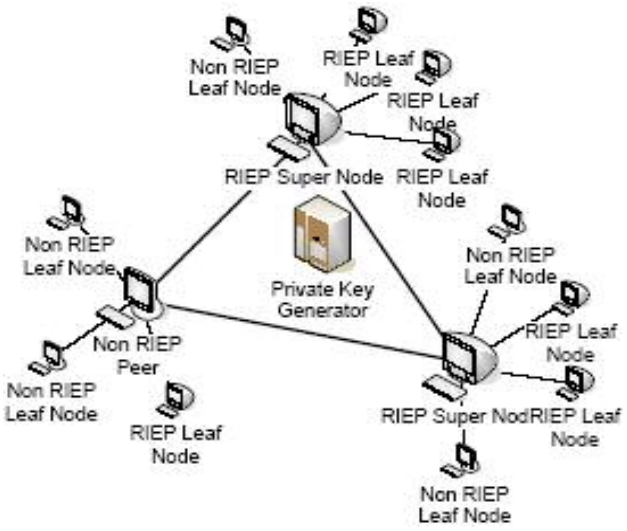
5 Implementation of RIEP in P2P Networks

As specified, the changes brought by proposed RIEP is applicable to all P2P file-sharing networks, despite their differences in design. While we have proved that RIEP successfully promotes peer accountability and defends against index poisoning attacks, there are some implementation details and limitations that need to be discussed.

5.1 RIEP Implementation Requirements

To maintain backward compatibility with existing P2P file-sharing networks, RIEP sends all its messages in plain text. The only changes made to the file index table are the 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.

We recognize that RIEP does put extra workload on each peer for signature signing and verification. However, since a typical peer is unlikely to publish more than a few dozen of files, such overhead is well within acceptable limits. Figure 3 shows the gradual transition from a generic, non-RIEP to a RIEP-enabled P2P network. During the transition, some peers may elect to adapt RIEP while others are reluctant to change.



FileID	PeerID	Publisher's Signature	PKG ID	PKG Signature
File0	RIEP0	S_{RIEP0}(File0)	PKG0	S_{PKG0}(RIEP0)
File1	Non-RIEP	N/A	N/A	N/A
File2	RIEP1	S_{RIEP1}(File2)	PKG1	S_{PKG1}(RIEP1)

(a) RIEP-enabled peers **(b) Shared file indexes among all peers**

Figure 3 RIEP protocol allow coexistence and gradual conversion of non-RIEP peers

5.2 Private versus Public IP addresses

RIEP uses a public IP address as part of its public key of peer in Fig.4. Our design decision is based on the following observation: most P2P file-sharing users are home users. They connect to the Internet via broadband connection from their ISP. In most cases, these users use firewall/router to establish home networks, and the P2P hosts are DHCP clients of the routers. Therefore, a P2P host's own IP address cannot be used to establish connection.

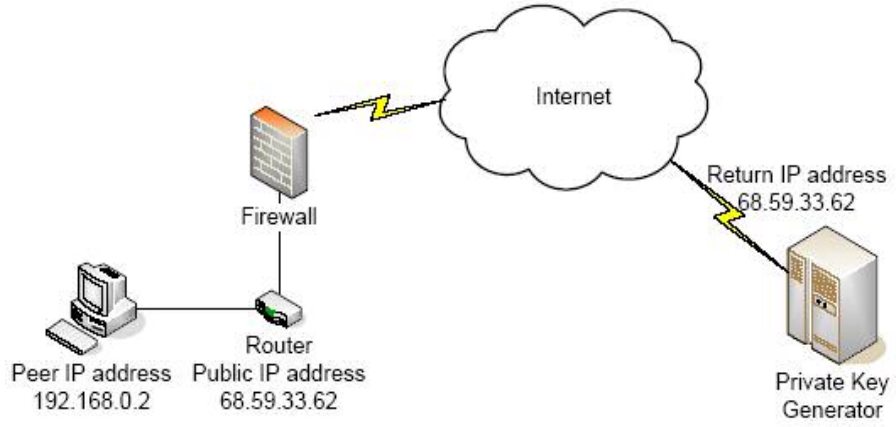


Figure 4 The problem of using private vs. public IP address for a peer behind a router. The public IP address seen by the PKG is used as part of the public key

A PKG can identify the public address of a peer from its private key request packet. Because the same address is used when the peer establishes connection with other peers, we use this address as part of the public key. As discussed in section 4, using public IP address

can help defend against a list of attacks. Using the entire endpoint as oppose to only the IP address help distinguish from peers when multiple peers are behind the same firewall, as many home networks have multiple computers that access Internet simultaneously.

5.3 Multiple Private Key Generators

In our previous discussion, we pointed out that for a n-node P2P file-sharing network, PKG's workload is O(n). To further lower PKG's workload, RIEP supports multiple PKG's. The PKG ID in *R* is used to identify individual PKG. When requesting private key, a peer can choose from a list of well known, trusted PKG's. A chosen PKG will sign the returning message with its private key; this signature is also part of *R*. A peer receiving the file index *R* would then use a verification function corresponds to PKG ID to verify peer accountability.

Any single PKG has the capability of signing all RIEP messages. If an attacker can masquerade itself as a valid PKG, it will bring down the entire peer accountability structure established by RIEP. Because of this threat, the identities of all PKGs must be further protected. We use the common public key infrastructure: each PKG must register its identity and its public key with a trusted certificate authority; and all peers will verify PKG identity with the certificate authority.

5.4 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 revoking, that are inherent to P2P file-sharing networks. These problems may impose some limits to apply the RIEP protocol. We discuss below a possible solution to handle the multiple router problem. The problem and solution are illustrated in Fig.5.

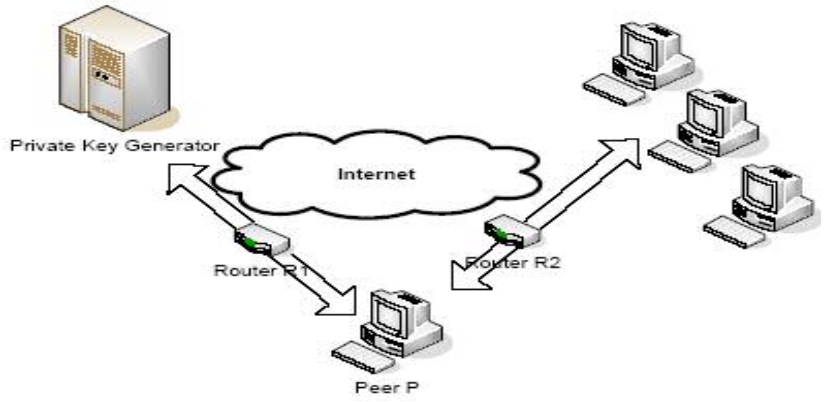


Figure 5 A peer behind multiple routers with static routing table will have problem to publish the file index under RIEP

In many organizations, there are multiple routers connecting the Intranet to the Internet. These routers most like to use fixed routing tables. The RIEP may stop a peer from being able to exchange index with other peers. The peer P is connected to two Internet gateway routers: R1 and R2. The Internet traffic towards the PKG goes through the router R1 and traffic to peers goes through the router R2. Because CA only sees the public IP address of R1, it will issue the private key accordingly. With this private key and IP address, peer P publishes the file indexes to other peers via router R2. This will create the confusion at the receiving peers.

Since the signatures match, the receiving peers accept the file indexes published by peer P. However, when they try to contact peer P via claimed endpoint, they are sending requests to router R1 instead of router R2 because 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 opening ports in router R1. This problem causes other peers to accept the published peers, but fails to establish the correct connections.

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

6 Performance of RIEP-enabled P2P Networks

We evaluate the performance of RIEP via simulation experiments. Because the RIEP applies to any P2P file-sharing network, we simulate a generic P2P network with one million peers. The simulation setting and performance metrics are described first. Then we report the simulation results in remaining subsections.

6.1 The RIEP Simulator and Performance Metrics

We assume the following behavior of simulated peers and attackers. Define a peer session as the time period between a peer joining and leaving the network. Peers randomly join and leave the network. Peer session length follows an exponential distribution with an average length equal to 30 minutes. The attacker launches attack by sending poisoned file indexes to all peers. By crawling through the neighbor list of each peer, it can reach almost all peers within 45 minutes.

Poisoned file index propagate via peer index query. When a peer is looking for a particular file, it sends lookup query to its neighbor. If one of the neighbors has poisoned file index, the peer will be infected with the poison. Every second, each peer that is infected with the poisoned index will attempt to connect to the victim with probability p. This probability represents the chances that a peer indeed attempt to download the poisoned file. In our simulation, the default value is set at $p = 0.001$.

Peers connect to the victim requesting for a file download, while the victim cannot provide such service to the peer. The average size of these requests is 256 bytes. The peer will attempt at most three connections to the victim; afterwards it will discard the file index as unreachable. The victim is a web server that allows keep-live connections. It will terminate any connection that has been idle for more than 2 minutes. Because the web server is NOT part of the P2P file-sharing network, there is no legitimate traffic from the peers during simulation.

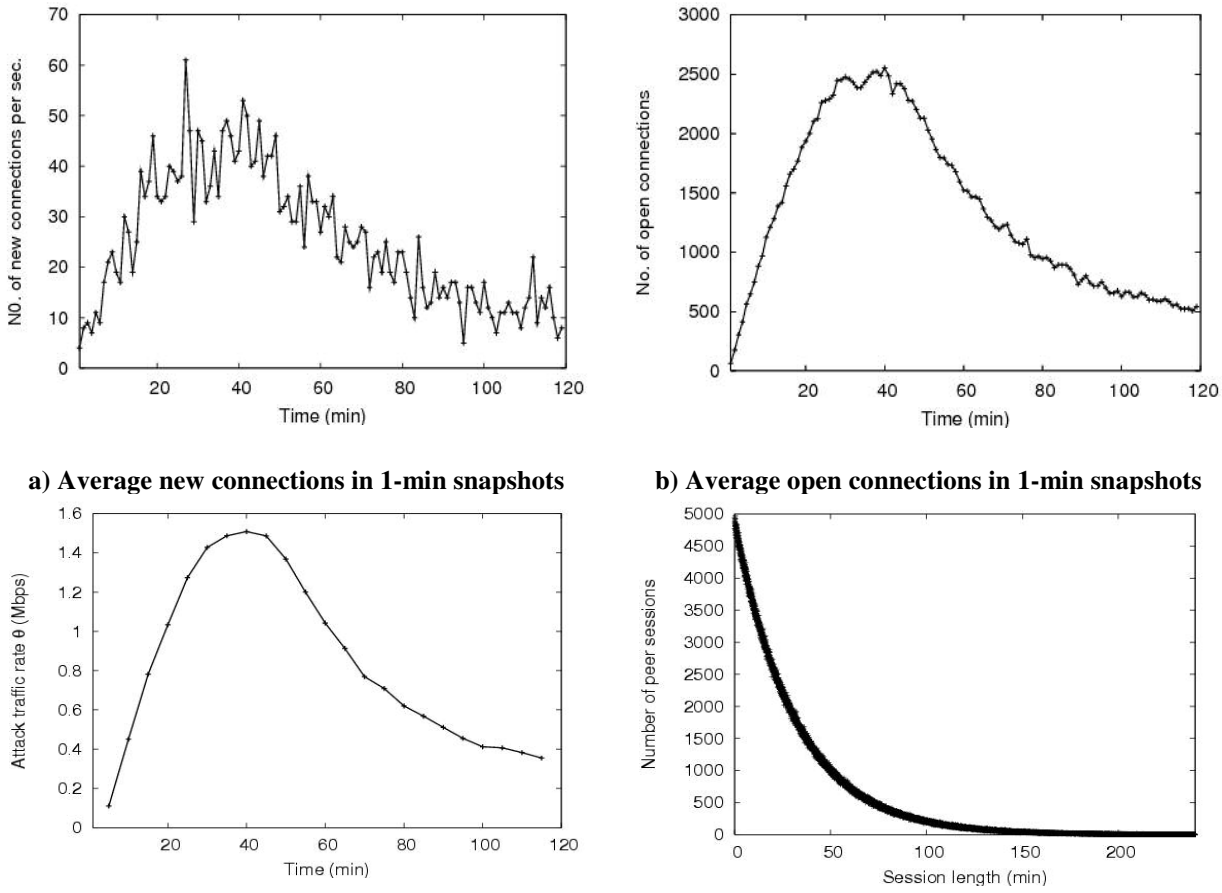
There is no bandwidth bottleneck at any peer. This assumption is based on the observation that most P2P file-sharing users are broadband connected. The bandwidth consumption for a file download request is ignorable. When a RIEP-enabled peer receives unsigned file indexes, it discards the index. This is the safest option under RIEP protocol, at the cost of lower content availability to those peers.

These behaviors are common among P2P file-sharing client software. Three matrices are of particular interest: *attack traffic rate, opened connections,* and *residue effect*. Attack traffic rate and opened connections represents the intensity of DDoS attacks. Residue effect is unique to this type of security threat, since the poisoned index will reside on peer's caches for a long time after the attacker left.

6.2 Effects of Index Poisoning Attacks

In a baseline study, we assume none of the one million peers is RIEP enabled. Therefore, an index-poisoning attack will achieve its maximum effect. Figure 6 reports our simulation results on index poisoning attacks without RIEP defense. Figure 6(a) plots the number of new connections to the web server. Each data point represents the average connection rate within 1-minute interval. This level of new connection request will very quickly saturate available ports on the server.

Figure 6(b) plots the average number of TCP open connections for 1-min intervals at the victim host. Very similar to Fig.6(a), these open connections keep increasing until it reaches a maximum of 2,500 when the attack window ends. After the attacker leaves the network, poisoned peers are still request file download from the victim, resulting in some residue open connections. Two hours after the attack, open connections are still very high, which seriously affects the ability of the web server to serve legitimate http requests.



a) Average new connections in 1-min snapshots

b) Average open connections in 1-min snapshots

c) Attack traffic rate in 5-min snapshots

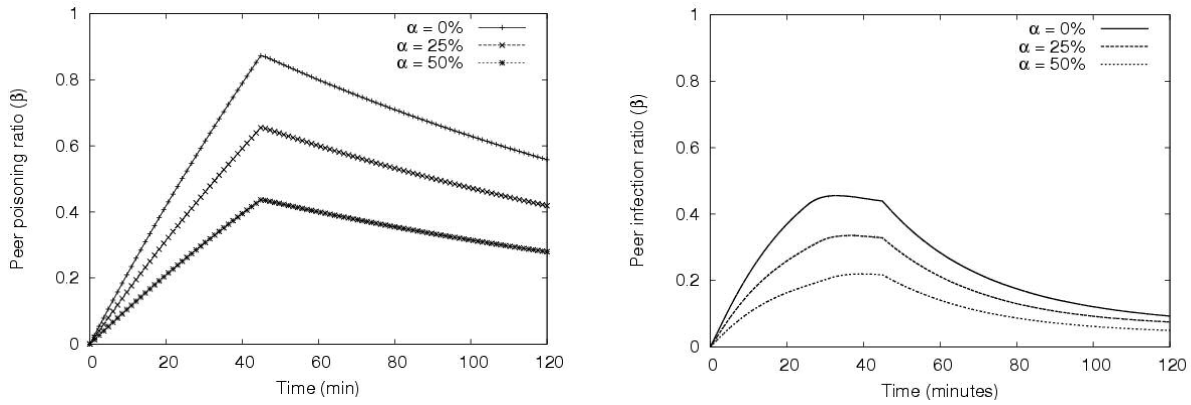
d) Connection length distribution

Figure 6 Effects of index-poisoning DDoS attacks on a simulated P2P network over 1 million of nodes, which are not RIEP-enabled, during the first 2 hours of the attack.

Figure 6(c) shows the attack traffic volume as observe by the victim. This curve is has the highest attack rate at 1.5 Mbps, similar in shape as Fig.6(b) except the scale and unit along the y-axis. This simulation result is very close to the real-life experiments reported by Naoumov et al[22]. These results have validated our simulation settings. Figure 6(d) reports that connection lengths are exponentially distributed. About 85% of the sessions lasted less than an hour.

Figure 7(a) plots the peer poisoning ratio θ using Eq.(9) and Eq.(11). These results are measured on a static P2P network, ignoring the effects of random peer joining or departure. Obviously, θ is lowered proportionally with the percentage of RIEP-enabled peers. The peak values appear at the same time (45 minutes after the DDoS attack) when the attack window T expired.

Figure 7(b) plots simulation results measured on a dynamic P2P network allowing random peer join and leave. For the same peer enabling ratio α, the peer poisoning ratio β is lowered significantly in the dynamic networks. In our experiment, the random joining and leaving of peers can reduce about 50% of the poising effects 30 minutes after the attack. This implies that our theoretical results reflect the worst-case scenario, giving an upper bound on the real-life index poisoning effects.



(a) Theoretical results on a static network **(b) Simulation results on a dynamic network**

Figure 7 Variation of the number of infected peers during and after the DDoS attacks on static and dynamic P2P networks over one million nodes

6.3 Results on RIEP Experiments

The RIEP protocol is designed so that peers can gradually migrate from non-RIEP client software to RIEP-enabled clients. This gradual transition is important for the protocol modification to be practical. In this section, we simulate 10%, 25%, and 50% of the peers apply RIEP to defend against index poisoning attacks.

A. Attack traffic rate and peer infection rate

Figures 8 reports the variation of attack traffic rate with time under different α values. The similar results on open connections and attack traffic rate prove that the intensity of index-poisoning attacks reduces with RIEP-enabled peers. This matches with the analytical

prediction in Sec.4. Because RIEP peers discard the unsigned file indexes, the secured peers are unavailable to the attacker. Therefore, the attack intensity drops quickly.

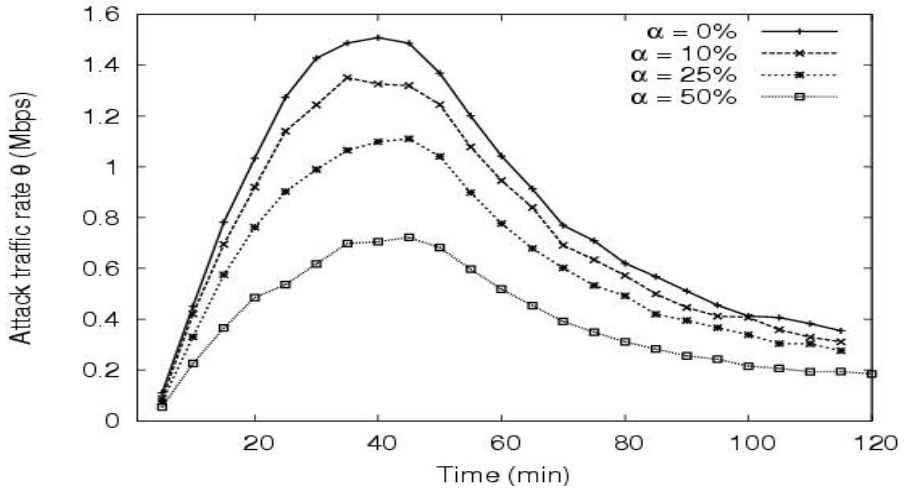


Figure 8 DDoS attack traffic surges under different levels of RIEP-enabled peers

Varying the attack speed leads to different attack window sizes. If the peer infection rate is high, the attack may use a smaller window. Intuitively, a fast attacker is much more efficient. Figure 9 plots the theoretical results on three attack windows values. These plots demonstrate clearly that the peer poisoning ratio θ increase sharply to a higher peak value of 95% for the smaller 15-min attack window. The growth rate and peak value decrease with the enlargement of the window size. With a 120-min window, the poisoning ratio drops to a peak of 70% and the rise and fall of the poisoning become slow.

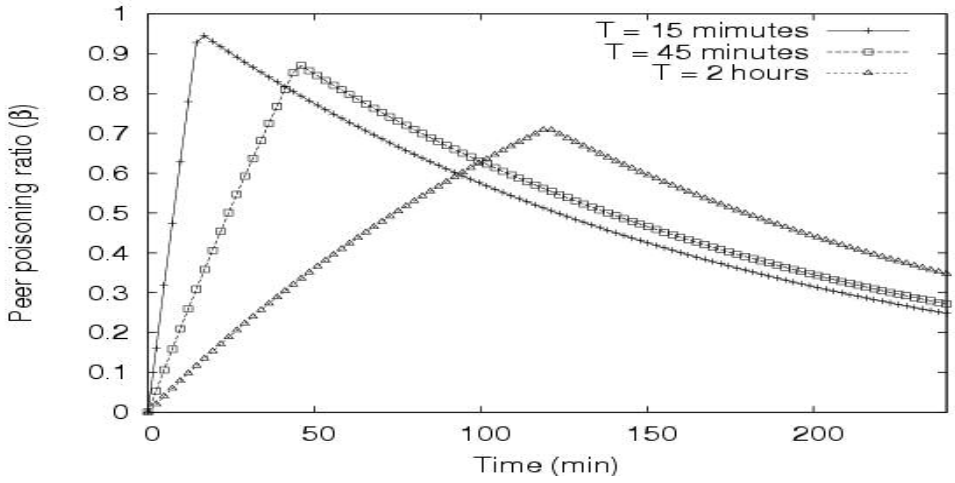


Figure 9 Effect of attack window size on the infection rate

B. Residue Effect after the Attack

Each of our simulations lasted 10 hours. Using attack traffic volume as a metric, we evaluate the residue effect of index poisoning attacks under different level of RIEP peer population. Residue effect refers to the fact that long after the attacker left the network, the victim is still under attack. Residue effect can be measured as the sustained attacking traffic rate after attacker left the network. Figure 10 shows our simulation results on the attack traffic rate θ in Mbps.

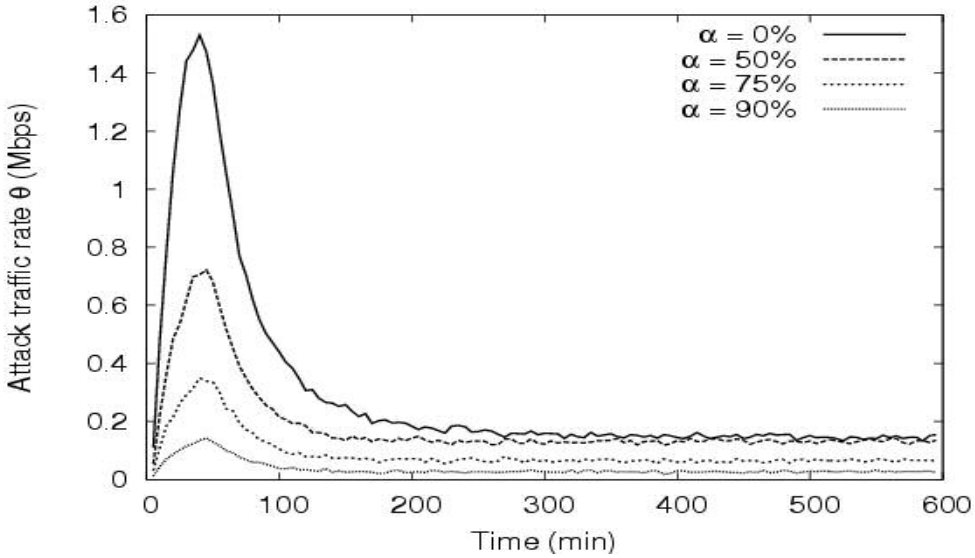


Figure 10 Effects on residue DDoS attack traffic under various percentages of peers that are protected by RIEP

The attack traffic rate reaches its peak 45 minutes after the attack initiated. Then the traffic rate subsides quickly to a stable low residue value. The base line case has the highest peak rate. The peak rate reduces inversely proportional to the α value. The attack traffic rate of both the baseline ($\alpha = 0$) and $\alpha = 50\%$ drops to 180 Kbps 4 hours after the attack. With 75% of the peers immune from index poisoning attacks, the residue attack rate is at 70 Kbps. If majority (90%) of the peers is RIEP enabled, the rate is further reduced to 30 Kbps.

7 Conclusions

P2P file-sharing networks are built on a large number of peer hosts running the same software. Anonymity of these peers is the key for the popularity of the P2P network. This feature is in direct conflict with any attempt to post user authentication in a P2P network. For this reason, current designs of P2P file-sharing networks are most vulnerable to index

poisoning DDoS attacks. Via index poisoning, an attacker can launch sustained DDoS flooding attacks on any host inside or outside of the P2P network.

To defend against index-poisoning DDoS attacks, we propose a new RIEP protocol to enforce the peer accountability. We developed analytical models to estimate the growth of infected peers under index-poisoning attacks using the RIEP defense, as compared with those cases without protection. Analysis proves that RIEP-enabled peers cannot launch index-poisoning attacks on unauthorized IP addresses. The scheme can also identify those peers who are attempting to launch the attacks.

This peer accountability 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 ability of choosing an external target. Via extensive simulation experiments, we demonstrate that RIEP can effectively reduce the attack traffic towards the victim. The residue attack traffic can be maintained very low, if majority of the peers are RIEP-enabled.

For further research, we suggest to integrate the defense techniques to prevent index poisoning with the content poisoning techniques. The integrated approach will benefit both DDoS defense and copyright protection in P2P file-sharing systems. Our continued efforts are heading in this direction. Other interesting challenges are to launch large-scale benchmark experiments to test the viability, scalability, and dependability of real-life P2P networks like the BitTorrent, KaZaA, eMule, Gnutella, Fasrtrack, and Freenet, etc. Standard P2P benchmark programs are very much in demand in the years to come.

Acknowledgements: This work was supported by NSF ITR Grant ACI-0325409 at USC Internet and Grid Research Laboratory. Discussions with Min Cai of the GridSec research team are greatly appreciated.

References:

[1] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies," in *ACM Computing Surveys*, vol. 36. New York, 2004, pp. 335--371.

[2] J. Baek, J. Newmarch, R. Safavi-Naini, and W. Susilo, "A Survey of Identity-Based Cryptography " *Proc. of Australian Unix Users Group Annual Conference 2004*.

[3] K. Berket, A. Essiari, and A. Muratas, "Pki-Based Security for Peer-to-Peer Information Sharing," *Proc. of the Fourth International Conference on Peer-to-Peer Computing (P2P'04) - Volume 00*, 2004.

[4] D. Boneh and M. K. Franklin, "Identity-Based Encryption from the Weil Pairing," *Proc. of the 21st Annual International Cryptology Conference on Advances in Cryptology*, 2001.

[5] CacheLogic, "Understanding the Impact of P2p: Identifying and Measuring Traffic," 2006, http://www.cachelogic.com/home/pages/understanding/identifying.php.

[6] J. C. Cha and J. H. Cheon, "An Identity-Based Signature from Gap Diffie-Hellman Groups," *Proc. of the 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography*, 2003.

[7] G. Chen and R. S. Gray, "Simulating Non-Scanning Worms on Peer-to-Peer Networks," *Proc. of the 1st international conference on Scalable information systems*, Hong Kong, 2006.

[8] A. Cheng and E. Friedman, "Sybilproof Reputation Mechanisms," *Proceeding of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, Philadelphia, PA, 2005.

[9] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," *International workshop on Designing privacy enhancing technologies: design issues in anonymity and unobservability*, Berkeley, CA, 2001.

[10] "Peer-to-Peer File Sharing Privacy and Security," in *Testimony before the House Committee on Government Reform*, May 15, 2003, http://www.cdt.org/testimony/030515davidson.shtml.

[11] J. R. Douceur, "The Sybil Attack," *First International Workshop on Peer-to-Peer Systems*, Springer Verlag, 2002.

[12] FaceTime Communications, Inc., "Impact Report for Second Quarter 2006 (April - June)," http://www.facetime.com/securitylabs/impactreport_06Q2.aspx.

[13] M. J. Freedman, E. Sit, J. Cates, and R. Morris, "Introducing Tarzan, a Peer-to-Peer Anonymizing Network Layer," *First International Workshop on Peer-to-Peer Systems*, 2002.

[14] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload," *Proc. of the nineteenth ACM symposium on Operating systems principles*, Bolton Landing, NY, 2003.

[15] S. Hazel and B. Wiley, "A Variant of the Chord Lookup Service for Use in Censorship Resistant Peer-to-Peer Publishing Systems," *First International Workshop on Peer-to-Peer Systems*, Springer Verlag, 2002.

[16] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells, and B. Zhao, "Oceanstore: An Architecture for Global-Scale Persistent Storage," *SIGARCH Comput. Archit. News*, vol. 28, pp. 190-201, 2000, 379239.

[17] J. Liang, N. Naoumov, and K. W. Ross, "The Index Poisoning Attack in P2p File-Sharing Systems," *Infocom*, 2006.

[18] P. Maniatis, T. J. Giuli, M. Roussopoulos, D. Rosenthal, and M. Baker, "Impeding Attrition Attacks in P2p Systems," *Proc. of the 11th workshop on ACM SIGOPS European workshop: beyond the PC*, Leuven, Belgium, 2004.

[19] D. Manini, R. Gaeta, and M. Sereno, "Performance Modeling of P2p File Sharing Applications," *FIRB-Perf Workshop on Techniques, Methodologies and Tools for Performance Evaluation of Complex Systems, 2005.*, Torino, Italy, 19 September 2005.

[20] T. Mennecke, "File-Sharing Population Level through October," November 14, 2005, http://www.slyck.com/news.php?story=991.

[21] C. Miguel, D. Peter, G. Ayalvadi, R. Antony, and S. W. Dan, "Secure Routing for Structured Peer-to-Peer Overlay Networks," in ACM *SIGOPS Operating System Review, V*ol. 36. New York, NY, 2002, pp. 299--314.

[22] N. Naoumov and K. W. Ross, "Exploiting P2p Systems for Ddos Attacks," *International Workshop on Peer-to-Peer Information Management (keynote address)*, Hong Kong, May 2006.

[23] S. Saroiu , K. P. Gummadi, J. D. Richard, S. D. Gribble, and H. M. Levy, "An Analysis of Internet Content Delivery Systems," ACM *Operating System Review*, Vol. 36, pp. 315--327, 2002

[24] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," *Proc. of CRYPTO 84 on Advances in cryptology*, Santa Barbara, CA, 1985.

[25] M. Srivatsa and L. Liu, "Vulnerabilities and Security Threats in Structured Overlay Networks: A Quantitative Analysis," *20th Annual Computer Security Applications Conference.* , 2004.

[26] D. S. Wallach, "A Survey of Peer-to-Peer Security Issues," *International Symposium on Software Security*, Tokyo, Japan, November 2002.

[27] H. Wen and D. Gu, "Authenticated Group Key Agreement with Admission Control," *Proc. of the 3rd international conference on Information security*, Shanghai, China, 2004.

[28] L. Zhou, L. Zhang, F. McSherry, N. Immorlica, M. Costa, and S. Chien, "A First Look at Peer-to-Peer Worms: Threats and Defenses," *4th International Workshop on Peer-To-Peer Systems (IPTPS'05)*, Ithaca, NY, Feb. 2005.

Biographical Sketches:

Xiaosong Lou received the B.S. degree in Electronic Engineering from Shanghai Jiaotong University, China in 1994. He has worked in Unisys and TFTI as system engineer for 10 years. In 2005, he received his M. S. degree in Computer Engineering from the University of Southern California. Currently, he is a Ph.D. candidate in USC Computer Engineering program.

His research interest covers the areas of peer-to-peer and Grid computing, poisoning detection, distributed content delivery, and semantic networks. He can be reached via the email address: xlou@usc.edu

Kai Hwang is a Professor of Electrical Engineering and Computer Science and Director of Internet and Grid Research Laboratory at USC. He received the Ph.D. degree from the University of California, Berkeley. An IEEE Fellow, he specializes in computer architecture, parallel processing, Internet and wireless security, Grid, cluster, P2P and distributed computing systems.

Dr. Hwang is the founding Editor-in-Chief of the *Journal of Parallel and Distributed Computing* published by Elsevier. He is also on the editorial board of *IEEE Transactions on Parallel and Distributed Systems.* He has published over 200 original scientific papers and 7 research books. Presently, he leads the GridSec project at USC in developing distributed defense systems against worms and DDoS attacks for trusted Grid, P2P, and Internet computing. Contact him at kaihwang@usc.edu or visit the web site http://GridSec.usc.edu/Hwang.html