DDoS Vulnerability of BitTorrent Peer Exchange Extension: Analysis and Defense

Majing Su[1], Hongli Zhang[1], Bingxing Fang[1], Xiaojiang Du[2]
1 School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{sumajing, zhl, bxfang}@pact518.hit.edu.cn
2 Dept. of Computer and Information Sciences, Temple University, Philadelphia, PA, 19122, USA
dux@temple.edu

***Abstract*—BitTorrent (BT) is a well-known Peer-to-Peer (P2P) downloading protocol and has been implemented in several versions. New features and extensions used to improve performance of BitTorrent systems also bring some security issues. In this paper, we analyze potential DDoS vulnerabilities of BT and its Peer Exchange extension. We show the ways of launching connection-exhausted DDoS attacks. Our experiments demonstrate these attacks are persistent and incur few costs for the attacker. By analyzing the main causes we find that both the defect of implement and the lack of trust and authentication mechanism are to blame, while the latter is critical. To defend against the DDoS attacks, we propose a score-based peer Reputation Exchange (REX) mechanism. Using REX, the score of a malicious peer is less than that of a good peer after several iterations, hence has less chance to be connected. REX makes it difficult to launch a DDoS attack and it can effectively mitigate the effect of the attack.**

***Index Terms*—BitTorrent, P2P, DDoS attack, peer exchange**

I. INTRODUCTION

BitTorrent (BT), one of the most popular Peer-to-Peer (P2P) applications, has been widely used for delivering large files to massive number of users. BT is a centralized P2P overlay network. In the original BT protocol, the central server, referred as tracker, is an important node as it helps peers sharing the same files (known as a "swarm") find each other when initializing downloads and maintain their status. Distributed Hash Table (DHT) has been added to the current design of BT systems to offer decentralization and robustness. Peer Exchange (PEX) [1], a new extension of the BT protocol, is proposed to ease the bottleneck problem of the tracker. PEX allows a group of peers in a swarm to get more downloaders by exchanging IP/Port list of newly connected or disconnected peers. This increases the speed, efficiency, and robustness of the BT protocol significantly. With the combination of DHT and PEX, BT can be trackless.

The rapid development of P2P causes security problems such as the Sybil attacks [2], [3], content pollution and poison [4], [5], resource exhaustion attacks [6], [7], privacy and piracy issues. In [8] and [9], the authors discuss the vulnerabilities of P2P systems and present the practicability and ways of launching a variety of attacks. The scalability and popularity of BT systems also provide opportunities for malicious peers to launch attacks, especially Distributed Denial-of-Service (DDoS) attacks. Sia [10] shows the possibility by deploying a DDoS attack in real world. Harrington *et al.* [11] and Defrawy *et al.* [12] discuss similar BT-driven DDoS attacks to any host. Their real-network experiments and simulations show the applicability, feasibility and severity of the attacks.

A number of mechanisms have been proposed to defend against the attacks in general P2P systems, including using trust [13], [19], reputation and authentication [14], [18]. However, applying these mechanisms to the current BT system requires additional components, complex implementation or massive information exchanges. Existing methods on defending the BT-driven DDoS attack include target-based blacklisting, malicious trackers detecting and disabling, behavioral or probabilistic-based anomaly detection [15]. These methods are hard to deploy and cause large overhead. To sum up, there is no efficient method to detect and prevent the DDoS attacks in the current BT systems.

In this paper, first we present several new vulnerabilities of the BT PEX. A malicious peer may launch DDoS attacks by exploiting these vulnerabilities. Then we propose a novel BT extension protocol that can defend against the attacks. We summarize our contributions in the followings:

1) We discover that the PEX has inherent vulnerabilities which can be exploited by malicious peers to launch a connection-exhausting DDoS attack. The attack can be launched on multiple hosts or BT swarms. To the best of our knowledge, this is the first work to report these security vulnerabilities of PEX. The security issues may also exist in other P2P systems that allow peers to exchange neighbor lists.
2) We analyze the causes of the attacks and find out that the underlying reasons are the weak authentication and anonymity of BT systems.
3) We propose Reputation Exchange (REX), a novel reputation-based mechanism to defense the DDoS attack. Analyses and simulations show that REX can mitigate the attack effectively. REX has low overhead, and it is easy to deploy and compatible with existing BT systems.

The rest of this paper is organized as follows. In Section II, we show the PEX vulnerabilities by protocol analysis and real-network experiments. In Section III, we present our approach to defend the DDoS attack. In Section IV, we evaluate the performance of REX. We conclude our work in Section V.

978-1-4577-2053-6/12/$31.00 ©2012 IEEE

II. New Vulnerabilities of the BT Peer Exchange

A. New Vulnerabilities of the BT PEX

Most of the current BT client software and 95% of peers support the PEX extension [1]. PEX has been typically implemented using one of three common extension protocols: AZ_PEX, UT_PEX and BC_PEX. In all types of PEX, a peer periodically sends PEX messages to other peers. The PEX message lists a group of newly added peers and a group of newly removed peers. At present, major BT clients such as Vuze and µTorrent support these protocols and are compatible with each other. Due to the large population of BT users, the first and most important condition of DDoS is satisfied. Our analysis of the vulnerabilities of BT and Peer Exchange includes the following two aspects:

1) Protocol Analysis of BT and PEX

BT especially PEX is not strictly defined and has various implementations such as µtorrent, Vuze and Transmission. The various implementations may not be consistent in terms of actions, errors, black-list mechanism, and so on. Peers don't verify remote identity during the entire download process. This was not considered either when PEX was designed. When receiving a peer exchange message, a node tries to connect to the added peers in the PEX message for faster downloading. It's suggested to BT client developers that [18]: (1) there should be no more than 50 added peers and 50 removed peers in any given PEX message; and (2) a PEX message should not be sent more frequently than once a minute. However, most clients tolerate violations of the above restrictions, which make it possible for a peer to rapidly broadcast itself in a swarm. Consequently, it is possible to launch a DDoS attack by broadcasting the target's address to the swarm, as shown in Fig. 1. The compatibility of various BT client software worsens the security situation.

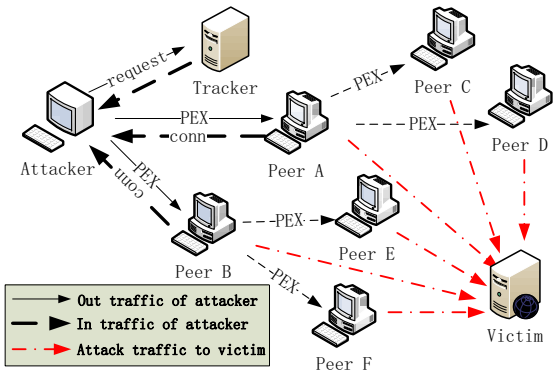

Figure 1. Sketch of a PEX-based DDoS attack

2) Measurement of User Behivors

In addition, by measurement, we find that some common features of BT clients can provide opportunities for adversaries to launch attacks. First, most BT clients will remain open before a 20-60 second time out if there is no response, and will try to reconnect 5 times if failed. These features make it possible for the attacker to achieve sustained attacks by sending the target address to peers only a few times. Furthermore, despite there is no data transmission, most clients still maintain connections to other peers for a period of time until timeout, and only a few clients will actively stop a connection. Hence, the attacker can keep in touch with these peers and continuously send PEX messages. Besides, certain BT clients allow a peer to simultaneously connect to a number of peers with the same IP address. This feature allows an attacker to connect to multiple ports of the same target (the same IP), which causes more damages to the target.

B. Approaches to Exploiting PEX

To launch a DDoS attack, the attacker must attract enough peers. This can be accomplished by registering its IP/Port(s) to the trackers and DHT, or pretending to be seeders of hot torrents. To increase the probability of being found, the attacker can also bind more addresses. PEX can be used in two types of DDoS attacks:

1) DDoS to Any Host in the Internet

Once established connections to peers, the attacker responds to the BT handshake and sends out PEX messages including the victims' IP/Port(s). The attacker doesn't need to download file data, which significantly reduces its own traffic. The peers then try to connect to the victims. Therefore, if the number of peers is large enough, the target will be overwhelmed with many BT connections from the peers. This causes a DDoS attack on the victims. Using PEX, the IP addresses of both the attacker and the victims are propagated to more peers. As a result, the attacker has more peers to launch the attack and cause more damages on the victims.

2) DDoS to Any Swarm of the BT

The attacker can also send PEX messages containing a large number of IP lists that do not exist or download the same file. A normal peer will try to connect to these IPs, but it will not find the file. If the attacker only wants to attack one swarm, it may generate the IP lists randomly. If the attacker wants to attack multiple swarms, it may send peer list of one swarm to another. This attack is also referred as index poisoning attack and can be combined with other attacks to cause more damages.

An attacker does not need high-performance hardware or large bandwidth to launch such attacks, because the attacker doesn't transfer the large files except the (short) BT handshake and Peer Exchange messages. An attacker only needs a public IP address to launch the attacks. The attack effect depends on whether the attacker can get a lot of downloading peers. For hot resources, the swarm has hundreds of thousands of peers, which is sufficient for an attacker to launch the DDoS attacks.

C. Experiments Showing Feasibility of the DDoS Attack

To demonstrate the feasibility and damage of the attack, we run real experiments in a controllable environment. In the experiments, there is an attacker node, which can register itself to trackers, accept BT connections from any peers in any swarms, but merely answer PEX messages and keep-alive messages. There are several victim nodes, which can accept connections, check request messages, count the number of BT connections, and then reset or timeout the connections. We deploy the experiments at several locations (Beijing, Harbin and Guangzhou) in China, and using different ISPs (China Unicom

and CERNET). All hosts have 100Mbps bandwidth. The experiment environment is capable of supporting a large number of concurrent connections. We choose 10 popular torrents (each torrent has more than 10 thousands peers) from the Internet and register the attacker's IP and ports to the trackers of these torrents. We launch the DDoS attacks in the controllable environment, and record the results for 46 hours. We sample the concurrent connections, incoming traffic and outgoing traffic of the attacker machine and targets' machines every 5 minutes.

In the experiments, we have control over the attacker and target machines. The attack traffic is generated by the real BT networks. Fig. 2 shows the number of connections of the attacker and the targets varying over time. The *x* axis is the time after the attack started. The *y* axis is the number of concurrent connections (in thousands) of the attacker and the average number of connections of ten targets (victim nodes). In the experiments, the attacker launched the attack for one hour and then stopped. Fig. 2 shows that the attack effect lasted for about 2 days, where the number of connections of the targets remained high. This is because the victims' addresses have been propagated to more and more peers, which all try to connect to the victims. We can also see that the number of concurrent connections of attacker is much less than the victims.

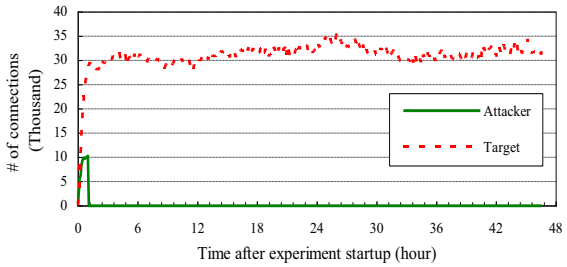

Figure 2. The number of connections of the attacker and victims

The PEX-based DDoS attack has several features. The attack doesn't expose the attacker as the actual attack traffic is from a large number of peers belonging to many different swarms. Furthermore, it is flexible to attack multiple hosts and service ports at the same time. Besides, as shown by the experimental results, this attack can last for a long time.

III. The Reputation Exchange Mechanism

The underlying cause of the DDoS attacks is the lack of authentication and validation mechanism in BT systems. The dynamic of BT peers and the compatibility between various BT clients make this security issue worse. In this section, we propose a score-based reputation mechanism that can stop false messages from spreading through BT networks. Specifically, we add reputation to the PEX message to monitor peers. Our Reputation Exchange (REX) mechanism consists of reputation score, reputation exchange protocol and protocol action.

A. Reputation Score

In REX, a reputation score is given to each peer. The score indicates whether a peer is trustable or not. Peers with high score are more trustable. First, we give some definitions. Suppose p_i and p_j are BT peers sharing file f. The length of f is L bytes, and peer p_i claims it has l bytes. p_i has downloaded B bytes from p_j and b bytes of which is valid. We define

$$CP(p_i, f) = l / L \tag{1}$$

$$VL(p_i, p_j, f) = (b / B)^{\omega} \tag{2}$$

where $CP(p_i, f)$ is the percentage of file f claimed by p_i. If p_i has the entire file, then $CP(p_i, f)=1$, and if p_i has none of the file then $CP(p_i, f)=0$. $VL(p_i, p_j, f)$ is the validity of data downloaded from p_j. ω is an impact factor. For the same proportion of valid data download, the smaller ω is, the larger VL is. ω is suggested to be between 10~20. If p_i hasn't downloaded any data from p_j, then $VL(p_i, p_j, f)=0$.

We use $CM(p_i, p_j)$ (given in (3)) to represent the quality of the communication channel between p_i and p_j. Denote t as the actual response time (the time interval between sending a BT message and receiving its response) of a request from p_i to p_j. Denote $\bar{t}$ as the average response time, we have:

$$CM(P_i, P_j) = \begin{cases} \exp(\eta(1 - t/\bar{t})), & t - \bar{t} \geq 0 \\ 1, & t - \bar{t} < 0 \end{cases} \tag{3}$$

where η is an impact factor and it is suggested to be between 0.005~0.5. If p_i couldn't establish a connection with p_j, then $CM(p_i, p_j)$ is 0.

Peer p_j's reputation score given by p_i is based on the following three parts.

Direct Score (DS). The direct score is used to evaluate the interaction between p_i and p_j, defined in (4), where $\alpha, \beta, \gamma (0< \alpha, \beta, \gamma <1)$ is the weight of CP, VL and CM, respectively, and $\beta > \alpha+\gamma$. If p_i has never connected with p_j, then $DS(p_i, p_k)=0$.

$$DS(p_i, p_j) = \alpha \times CP(p_j, f) + \beta \times VL(p_i, p_j, f) + \gamma \times CM(p_i, p_j) \tag{4}$$

Indirect Score (IS). p_j's IS comes from neighbors of p_i via reputation exchange mechanism. The IS of p_j is defined in (5), in which, $S(p_k, p_j)$ is the reputation score of p_j given by p_i's neighbor p_k, $S(p_i, p_k)$ is the reputation score of p_k given by p_i. If p_j is from the tracker or added by the user, then $IS(p_i, p_j)=1$.

$$IS(p_i, p_j) = \sum_{k=1}^{K} \frac{S(p_i, p_k) \times S(p_k, p_j)}{K} \tag{5}$$

Historical Score (HS). The historical score is the average of the reputation scores that p_i gives to p_j in the past. In (6), N is the number of times p_i computes reputation score of p_j, and S is given in (7).

$$HS_N(p_i, p_j) = \begin{cases} \dfrac{HS_{N-1}(p_i, p_j) + S_{N-1}(p_i, p_j)}{2}, & N \geq 2 \\ 1, & N \leq 1 \end{cases} \tag{6}$$

Reputation Score (RS). The reputation score is given in (7), where a, b, c is the weight of the three score, respectively. $S(p_i, p_j)$ is a decimal number between 0 and 1. We use two constants τ and $T (0<\tau \leq T<1)$ to bound $S(p_i, p_j)$. If $S(p_i, p_j)< \tau$, $S(p_i, p_j) = \tau$. If $S(p_i, p_j)> T$, then $S(p_i, p_j)= T$.

$$S(p_i, p_j) = a \times IS(p_i, p_j) + b \times DS(p_i, p_j) + c \times HS(p_i, p_j) \tag{7}$$

This reputation mechanism, to some extent, is like the credit evaluation mechanism in our real life. Indirect score is the credit

of p_j that p_i heard from others, and direct score is p_i's judgment of p_j from the interaction. Historical score is the impression accumulated during past contacts. All of the information is used to compute a new score of the peer.

B. The Reputation Exchange Protocol

1) Protocol Format

The REX protocol follows the extension format defined in [16], as shown in Fig. 3. The name of this extension is 'rep_ext'. The payload of the reputation exchange message is given in Table I. All items are organized as a Bencoded dictionary, and all items are optional. BT clients that do not support it can ignore such message.

Figure 3. Reputation Exchange protocol format

TABLE I. PAYLOAD OF REX PROTOCOL

Items	Description
added	Peers newly added. A string consisting of multiples of 6 bytes. First 4 bytes are the IPv4 address and last 2 bytes are the port number. All in network notation. This is the same as PEX.
added.rep	Reputation scores of added peers. A string consisting of multiples of 4 bytes. The number of 4 bytes is the same as the number of added peers. All in network notation.
added6	Peers newly added. 18 bytes IPv6 address. All in network notation.
added6.rep	Format is the same as added.rep
dropped	Peers newly dropped or in blacklist. Format is the same as added.
droped.rep	Reputation scores of dropped peers. Format is the same as added.rep
dropped6	Peers newly removed. Format is the same as added6.
dropped6.rep	Format is the same as dropped.rep

2) Protocol Actions

A peer supporting REX maintains a peer reputation score list, including DS, IS, HS and RS. This list is updated when sending or receiving a new REX message. Some scores may be deleted when they are expired or the list is too large. Next we'll discuss the protocol workflow.

When two peers p_i and p_k have established BT connections with each other, if p_k connects to a new peer p_j, p_k calculates $S(p_k, p_j)$ and then send the score to p_i. Once p_i received the REX message, p_i computes $S(p_i, p_k)$ and $IS(p_i, p_j)$, then updates $HS(p_i, p_k)$ using (6). p_i computes $S(p_i, p_j)$ when p_i wants to connect to p_j. If $S(p_i, p_j)$ is more than the trust threshold, p_i will connect p_j; otherwise, p_i will not connect p_j. If p_i has more than one peer, p_i connects to the peer with the highest score first. Once p_i finishes an interaction with p_j, p_i calculates $DS(p_i, p_j)$ based on the interaction, which is important for p_i to decide whether to introduce p_j to its neighbors.

In practice, BT developers and users may define the interval for exchanging reputations and the trust threshold. One principle is never send the REX message too frequently.

IV. PERFORMANCE EVALUATION

In this Section, first we discuss that the score of a malicious peer is less than that of a good peer after a few iterations, hence has less chance to be connected. Then, we evaluate REX via simulations and analyze its overhead.

A. Theoretical Analysis

Fig. 4 shows part of a logic topology of a BT swarm, where M is a malicious peer who doesn't upload correct data; P_1, P_2…P_n, R are normal peers. P_1…P_{n-1} get information about R and M from trackers and have transferred data with them for a period of time. Then P_1…P_{n-1} pass the reputation of R and M to P_n by using REX. Assume P_n has never connected to R and M before. We have the following lemmas.

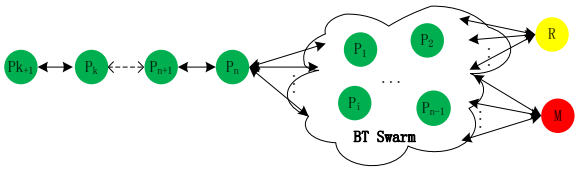

Figure 4. Part of a logic topology of a BT Swarm

Lemma 1. When P_n received REX message about R and M from P_1...P_{n-1}, if $\beta > \alpha+\gamma$, then $S(P_n, R) > S(P_n, M)$.

Lemma 2. If $S(P_n,R)>S(P_n,M)$, then $S(P_{n+1},R)>S(P_{n+1},M)$.

Lemma 3. If $S_t(P,R)>S_t(P,M)$, then $HS_N(P,R)>HS_N(P,M)$, t=1,2,…N-1.

Lemma 4. Consider a peer sequence, P_i, P_{i+1}, … P_m, where P_i tells the reputation of its connected peers R and M to P_{i+1}, P_{i+1} then tells the reputation to P_{i+2} … If m is large enough, even if $S(P_i,R)<S(P_i,M)$, there exists a K, when $n>K$, $S(P_n,R)>S(P_n,M)$.

Lemma 1~3 state that in the initial period of a swarm, the reputation score of a new peer depends on the indirect scores and the direct score from the connected peers. If $\beta > \alpha+\gamma$, no matter how much computation power or bandwidth the malicious node has, or how many files it declares, as long as it doesn't transfer valid file blocks to normal peers, its reputation score is less than that of a normal peer (e.g., node R). The gap between $S(P,R)$ and $S(P,M)$ is accumulated by historical scores. Lemma 4 states that despite some peers give a malicious node high scores and a normal peer low scores by mistake or on purpose, the scores will be corrected by other normal peers after several iterations. The proof also points out that the number of iterations has a logarithmic relationship with the difference between the RS of R and M. Even if the difference is large, only a few normal peers can correct the scores.

We also can infer from Lemma 4 that due to the score transfer among peers, an honest peer will get high reputation score and a malicious peer will get low reputation score. Hence, it is difficult for a malicious peer to launch DDoS attacks because it will get low score and would not be connected by normal peers after some time.

B. Simulation Experiments

To evaluate our mechanism, we simulate DDoS attacks by modifying the BT module of OMNet++ [17] with and without REX. There is a tracker used to help peers find each other. A malicious peer and a target do the same as the experiments in section II-C. Due to hardware limitation, we only simulate a swarm with 100 normal peers for 60 minutes. Every 5 minutes,

the system sends out REX messages that contain the peers' addresses of the top10 scores. We sample the connection number of the target every 1 minute. The parameters and result are given in Fig. 5. As we can see, when REX is used, the number of connections to the target drops significantly after 25 minutes (4~5 iterations). This means that REX effectively limits the DDoS connections. On the other hand, without REX, the number of connections to the target increases over time. The simulation results show that our REX mechanism is very effective in defending the DDoS attacks.

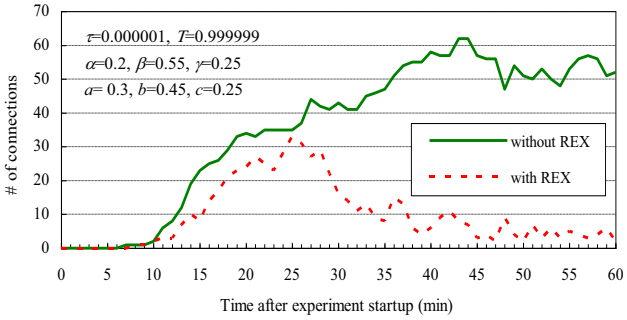

Figure 5. The number of connections to the target over time

C. Overhead

Storage Overhead - Each peer maintains a reputation score list (RSL) of other peers, and the storage overhead of each peer is 22*N*, where *N* is the number of items in the RSL. The size of each score (DS, IS, HS and RS) is 4 bytes. Each peer is identified by its IP and port #, which are 6 bytes (18 bytes if IPv6). For instance, caching 1000 peers requires only about 22KB of memory. If RSL goes too large, items can be moved to the disk or deleted using least-recently-used algorithm.

Bandwidth Overhead - Compared to PEX, REX only introduces 4 bytes additional traffic overhead for each peer in one REX message. If we follow the suggestion in [18], only 56 bps extra bandwidth is required.

Computation Overhead - The scores of each peer are computed only once either before sending or after receiving REX messages, and when initializing or closing connections. All of these actions are not very frequent, and a standard PC can do these.

V. Conclusion

In this paper, we analyzed the vulnerabilities of the BT Peer Exchange Extension protocol. Then we described two scenarios where an attacker can launch DDoS attacks by exploiting the vulnerabilities. Our real-network experiments showed that a malicious peer can launch the DDoS attacks to multiple targets over a long period. The weak authentication of BT systems is the main cause of the attack. We designed a score-based peer Reputation Exchange (REX) mechanism to defend against the DDoS attack. REX is easy to implement and compatible with current BT systems. Our simulation results show that the REX mechanism is very effective in defending the DDoS attacks.

Acknowledgment

This work was supported by the National Basic Research Program of China under Grant No. 2011CB302605, 2007CB311101, the National High-Tech Development 863 Program of China under Grant No. 2010AA012504, the National Natural Science Foundation of China under Grant No. 60903166 and 61173145 the Fundamental Research Funds For Central Universities under Grant No. HIT.NSRIF.2010041, and by the US National Science Foundation under grants CNS-0963578, CNS-1022552 and CNS-1065444.

References

[1] D. Wu., *et al.*, "Understanding Peer Exchange in BitTorrent systems," in *10th IEEE Int. Conf. Peer-to-Peer Computing*, 2010 © IEEE. doi: 10.1109/P2P.2010.5569967

[2] J. R. Douceur, "The Sybil attack. peer-to-peer systems," *Peer-to-Peer Syst.*, vol. 2429, pp.251-260, 2002.

[3] A. Singh, *et al.*, "Eclipse attacks on overlay networks: Threats and defenses," in *Proc. 25th IEEE Int. Conf. Comput. Commun.*, 2006 © IEEE. doi: 10.1109/INFOCOM.2006.231

[4] J. Liang, *et al.*, "Pollution in P2P file sharing systems," in *Proc. 25th IEEE Int. Conf. Comput. Commun.*, Miami , FL, 2005, pp. 1174-1185.

[5] J. Liang, *et al.*, "The index poisoning attack in P2P file sharing systems," in *Proc. 25th IEEE Int. Conf. Comput. Commun.*, 2006 © IEEE. doi: 10.1109/INFOCOM.2006.232

[6] N. Naoumov and K. Ross, "Exploiting P2P systems for DDoS attacks," in *Proc. 1st Int. Conf. Scalable Inform. Syst.*, Hong Kong, 2006 © ACM. doi: 10.1145/1146847.1146894

[7] M. Steiner, *et al.*, "Exploiting KAD: possible uses and misuses," *SIGCOMM Comput. Commun. Review*, vol. 37, no. 5, pp. 65-70, Oct., 2007.

[8] D. S. Wallach, "A survey of peer-to-peer security issues," in *Proc. 2002 Mext-NSF-JSPS Int. Conf. Software Security: Theories and System*, Tokyo, Japan, 2003, pp. 42-57.

[9] G. Gheorghe, *et al.*, "Security and privacy issues in P2P streaming systems: A survey," *Peer-to-Peer Networking and Applications*, vol. 4, no. 2, pp. 75-91, Jun. 2011.

[10] K. C. Sia, "DDoS vulnerability analysis of BT protocol," *UCLA Tech. Report*, avaliable at: http://netdrive.montclair.edu/~fogelconcej1/cs239spring06.pdf, 2006.

[11] J. Harrington, *et al.*, "A BT-driven distributed denial-of-service attack," in *Proc 3rd IEEE Int. Conf. Security and Privacy in Commun. Networks and Workshops*, Nice, France, 2007, pp. 261-268.

[12] K. E. Defrawy, *et al.*, BotTorrent: Misusing BT to launch DDoS attacks [Online]. Available: http://www.minasgjoka.com/papers/BotTorrent.pdf

[13] H. F. Yu, *et al.*, "DSybil: Optimal Sybil-resistance for recommendation systems," in *Proc. 30th IEEE Symp. on Security and Privacy*, Oakland, CA, 2009, pp. 283-298.

[14] M. A. Wang, *et al.*, "An adaptive and robust reputation mechanism for P2P network," in *Proc 2010 IEEE Int. Conf. Commun., 2010* © IEEE. doi: 10.1109/ICC.2010.5502541

[15] M. P. Barcellos, *et al.*, "Protecting BitTorrent: design and evaluation of effective countermeasures against DoS attacks," in *Proc 27th IEEE Symp. Reliable Distributed Syst.*, Napoli, Italy, 2008, pp. 73-82.

[16] *BitTorrent Extension Protocol* [Online]. Avaliable: http://www.rasterbar.com/products/libtorrent/extension protocol.html.

[17] K. Katsaros, *et al.*, "A BitTorrent module for the OMNeT++ simulator," in *Proc* 2009 *IEEE Int. Symp. Modeling, Anal. & Simulation Comput. and Telecommun. Syst.*, 2009 © IEEE. doi: 10.1109/MASCOT.2009.5366131

[18] *BitTorrent Peer Exchange Conventions* [Online]. Avaliable: http://wiki.theory.org/BTPeerExchangeConventions

[19] P. Shah and J.-F. Paris, "Incorporating trust in the bittorrent protocol," in *2007 Int. Symp. Performance Evaluation of Comput. and Telecommun. Syst.,* San Diego, CA, USA, 2007.

[20] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks," In *Proc. 12th Int. Conf. World Wide Web*, 2003© ACM. doi: 10.1145/775152.775242