A STUDY OF INDEX POISONING IN PEER-TO-PEER FILE SHARING SYSTEMS

Quan Yuan, Aaron Little, Maggie Kabore and Youssouf Kabore

Department of Math and Computer Science,University of Texas of the Permian Basin, Odessa, TX 79762

ABSTRACT

P2P file sharing systems are the most popular forms of file sharing to date. Its client-server architecture attains faster file transfers, however with its peer anonymity and lack of authentication it has become a gold mine for malicious attacks. One of the leading sources of disruptions in the P2P file sharing systems is the index poisoning attacks. This attack seeks to corrupt the indexes used to reference files available for download in P2P systems with false data. In order to protect the users from these attacks it is important to find solutions to eliminate or mitigate the effects of index poisoning attacks. This paper will analyze index poisoning attacks, their uses and solutions proposed to defend against them.

KEYWORDS

Peer-to-peer, index poisoning, ethical usage, attack prevention

1. INTRODUCTION

The Peer-to-Peer (P2P) systems started receiving attention with the introduction of Napster (the first popular file sharing system) in 1999. Since then P2P systems have grown in usage and have become one of the most important applications in the Internet, by many measures. Today, lots of users share various types of files, such as MP3 songs, entire albums, movies, documents, images, software, and games, through P2P systems (including BitTorrent, eMule, isoHunt, etc.). Meanwhile, P2P networks have also been subject to numerous attacks, and among them is index poisoning. An index poisoning attack deliberately advertises a large quantity of invalid peer information of some desired content that does not correspond to any existing content, IP address or available port number. That results in a peer spending a great amount of time on connecting to invalid peers where it fails to establish a connection, hence reducing the Quality of Service (QoS). On the other side, existing research demonstrates that index poisoning can serve not only as an attack strategy but also as a defence system in some specific situations. Therefore, how to use index poisoning and how to resist attacks caused by index poisoning in P2P systems, arouse peoples more attention than ever before.

We observe that there are several challenges on index poisoning in P2P file sharing systems, including: 1) as a attack strategy, index poisoning can be used for various attacks, such as P2P pollution, DDoS (distributed denial-of-service), people have to design different schemes to resist

DOI: 10.5121/ijci.2014.3602

hose attacks; 2) since P2P systems have different overlay structures, index poisoning prevention methods should be adjusted accordingly to adapt the various P2P structures; 3) besides as an attack strategy, index poisoning can also be utilized in an ethical way, to resist other attacks. How to apply it rationally is another interesting research topic. Based on the above challenges, researchers propose various schemes to push the investigation on the index poisoning in P2P systems. For example, several index poisoning prevention schemes are proposed for BitTorrent, distributed hash table (DHT), and Botnet, where other techniques such as cryptography, reputation system are applied. Also, researcher employ index poisoning to protect copyright in P2P system, and disrupt the Botnet system, with an ethical intention.

In this paper, we investigate the existing work on index poisoning, and present a survey of it in P2P systems. Specifically, we summarize there are two folds of the applications of index poisoning: attacks and ethical usage. Regarding to the attacks caused by index poisoning in P2P system, we divided the current prevention methods into two categories: proactive and reactive. Also, we compare and analyze the advantages and disadvantages of the methods in each category. To the best of our knowledge, this is the first work which provides a systematic survey for analyzing the index poisoning in P2P systems.

The remainder of this paper is organized as follows. Section 2 introduces the background we use in this paper. Section 3 discusses the ethical uses and unethical applications of index poisoning in the P2P networks. We then present an overview of the current prevention methods for the attacks caused by index poisoning in Section 4. Section 5 concludes the paper.

2. BACKGROUND

To state our survey work in a clear way, we first introduce some related background topics in this paper, including P2P Systems, and index poisoning.

2.1. P2P Systems

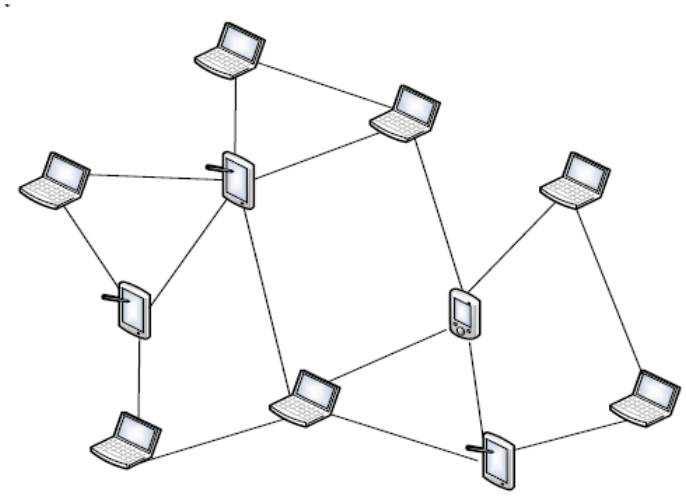

Figure 1.a Unstructured P2P network

Different from the centralized client-server architecture, P2P systems [16] are decentralized, where tasks or workloads are partitioned between peers. In another word, those peers form the P2P systems. Generally, peers are equally privileged, and collaborate with each other by contributing a portion of their resources, such as disk storage and network bandwidth to the whole P2P system. Note that each peer is both supplier and consumer of resources. In terms of topologies, P2P systems can be divided into two categories: unstructured and structured. In the unstructured P2P systems (Gnutella, Gossip, and Kazaa), peers randomly connect with each other to form the overlay network, without any particular structure constraint, as shown in Figure 1.a. Such systems are relatively easy to build up, but it is challenge to manage. For example, in Gnutella [15], a search query has to be flooded through the network to find who has the requested data, which is inefficient. On the other hand, in the structured P2P systems, peers have to follow a specific protocol to get organized into a specific structured overlay topology. Compared to the unstructured P2P systems, the structured systems need to consume more resources to maintain the overlay topology, but their resource searching is more efficient. Figure 1.b shows a structured P2P systems, chord [18], where nodes and file keys have an unique identifier using consistent hashing. Due to the consistent hashing, nodes and keys are uniformly distributed in an identifier circle, which also makes nodes join and leave the network without disruption. Each node has a successor and a predecessor. The successor to a node is the next node in the identifier circle in a clockwise direction, while the predecessor is counter-clockwise. Each node only needs to know how to contact its current successor node for look up. Queries for a given identifier of a keyword could be passed around the circle via these successor pointers until they encounter a pair of nodes that straddle the desired identifier, and the second in the pair is the node the query maps to.

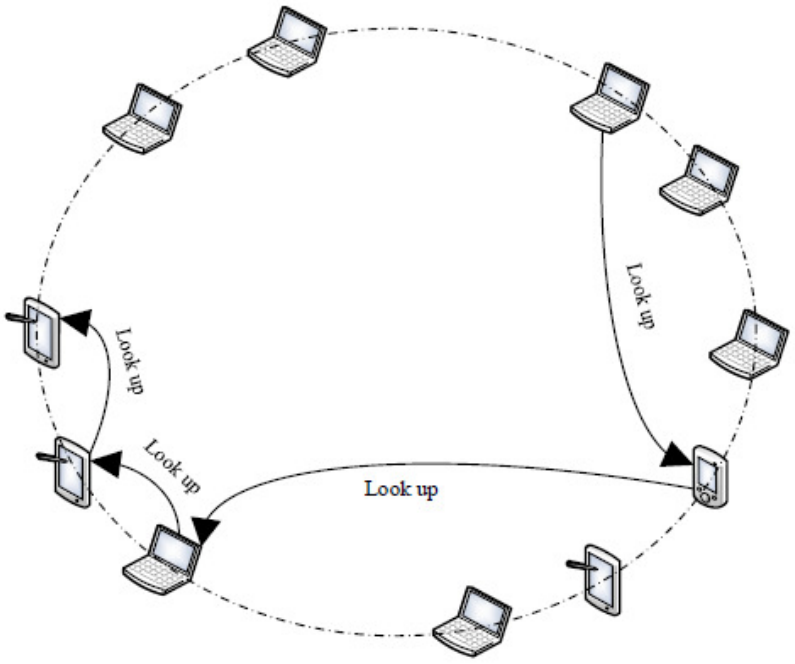

Figure 1.b Structured P2P network

One of the most popular applications of P2P systems is file-sharing, where users can download files from multiple nodes, instead of one. A shared file could be a song, video, etc.. Each file has

a file title, and it can have many copies in the network. This is because users download the same file from each other, which generates multiple copies of identical file in the P2P filesharing system. To let other users know the shared files, each node advertises to the distributed index about those files. For a specific shared file, the advertised information includes the file title, the location of the file, and the associated keywords of the file. A distributed hash table (DHT) [18] is one popular option for index information distribution, where a variant of consistent hashing is used to assign ownership of each file to a particular peer. In the DHT, (key, value) pairs are stored, and users can retrieve the value associated with a given key. Generally, a key could be a file name, and the value is the location information, such as IP address and port number of the node who is sharing that file. When a user wants to query for a specific file, he can perform a keyword search by typing in the keywords associated with the file title. Such information is sent to the distributed index, such as DHT. Then it is responded with the IP address and port number, where it can download the matching copies.

2.2. Index Poisoning

An index poisoning attack intentionally advertises a large quantity of false index information of the target files which does not correspond to any existing IP address or available port number [25]. As a result, when a user queries those target files, he may get that poisoned index, and try to connect the referenced IP address who actually is not a provider. As shown in figure 2, a file rovider first publishes the index of the shared file A, which usually includes the shared file name, provider's address information, to the index system in P2P networks. When a user wants to download file A, it has to send a file request to the index system, which will search for the matched index based on the keyword in the file request, and respond with the provider's address information. Once the user receives the provider's address information, such as IP and port number, it connects to the provider for downloading. An index poisoning attacker can jeopardize such process by injecting dummy index of file A into the index system. The dummy index includes the file name, and the address information of a dummy provider, who cannot upload file A. When the index system response with the dummy index to the user's file request, the user will try to connect those dummy providers but cannot download the desired file A.Moreover, index poisoning attacks can make the user repeatedly select advertised copies with a non-existent provider. That results in a peer wasting a great amount of time connecting to invalid peers, which eventually reduces the network QoS [4]. It is found that on certain P2P networks, a successful oison index attack can reduce the ability to download to 0.004% [22].

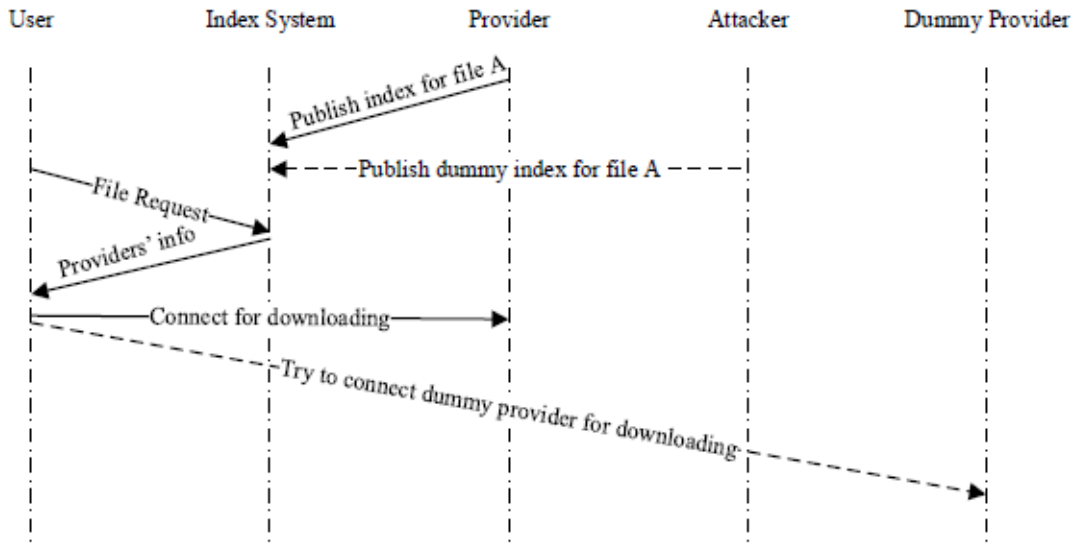

Figure 2 Index Poisoning Process

Poisoning is possible when the index does not efficiently authenticate the received advertisements, especially it does not authenticate whether a file copy is available at the referenced location (IP address and Port number) as claimed in the advertisement. Note that authenticating advertised files is quite challenging due to the open nature of the P2P systems. For example, a song may be encoded many different ways (.mp3, .wav, mov, etc.) and then within each of these encodings a song may have different versions. For a popular song file, a file-sharing system may have thousands of different versions. Because of the large number of different versions of one file title, it is difficult to apply a standardized way to authenticate every shared file's index information against index pollution.

3. APPLICATIONS

Even though index poisoning is considered as an attack method, it also can be used in an ethical way. In this section, we summarize the applications of index poisoning, which can be divided into two categories: Attacks and Ethical Usage.

3.1. Attacks

Index poisoning can be used as one form of P2P Pollution [4, 5]. P2P pollution is a term used to encompass the various methods used to block or disrupt P2P networks, such as content pollution. Content pollution attacks replace some of the file content with incorrect content, then shares it in high volume [13]. Also, servers must make the corrupted content available in high bandwidth to attract peers and will demand a lot of computing power from the numerous requests that will result. Index poisoning simply takes advantage of the lack of verification against modified file indexes to cause significant disruption. Compared to other P2P pollution attack methods, the advantage of index poisoning is that it requires fewer resources and bandwidth.

Note that one of the shortcomings of index poisoning lies in the robust architecture of the P2P networks. Therefore, to carry maximum disruption at an optimal cost, some P2P pollution attacks may combine the content pollution and index poisoning, considered as combined pollution. In [5], authors proposed an attack approach which hybrids the index poisoning and fake-block-attack, and apply that approach in BitTorrent. It not only hinders the establishment of peers' connections by index poisoning, but also tampers the data transmission by fake-blockattack. Specifically, the attacker firstly advertises large quantity of peer information into the tracker when it starts to pollute a BitTorrent swarm. Different from the index pollution, the peer information advertised in the tracker is not only the invalid peer information but also the peer information of the attacker. After that, when a good peer joins this swarm to start download, it firstly gets the peer information of the other benevolent peers from the tracker. That process will be affected by index poisoning, which means the downloading peer will find it difficult to establish connection with benevolent peers. However, a few downloading peers may overcome the effect of index poisoning and connect to some available benevolent peers by constantly trying to connect to them. Since the attacker is also considered as a benevolent peer at tracker, some of the downloading peers will actually connect to the attacker for download, but the downloaded data from the attacker is fake.

Another possible attack caused by index poisoning is Distributed Denial of Service (DDoS) attacks on unsuspecting victims. In [9], authors discuss that index poisoning can be employed to cause a DHT to be the source of a powerful DDoS attack against any arbitrary host. Note that in a DHT, a source node sharing content advertise to the DHT information about the content as well as the its IP address and port number. Thus, an attacker can use the poisoning attack in a completely different context, which is attacking a targeted host, instead of a specific file title. If for a number of popular titles the attacker inserts into the DHT numerous poisoned records pointing to a targeted host, the users that want to download those titles will be directed to the targeted host for downloading. Those users will then repeatedly send requests to the targeted host. In a large scale P2P system, those numbers of requests may generate a successful DDoS attack.

3.2. Ethical Usage

Besides considered as an attack method, index poisoning can also be carried out for ethical purposes to resist other attacks.

One popular ethical usage of index poisoning is to defend against the distribution and download of the copyrighted materials. Due to the massive sharing amount in P2P systems, any file could be rapidly and effectively distributed, including unauthorized file content. To avoid the huge losses, the "copyright industry" (such as the music, film, television, gaming, and book publishing industries) has a significant desire to prevent the unauthorized distribution of content in P2P systems. Index poisoning can be utilized for that with less resources and bandwidth, compared to other pollution methods. Also, network monitors and the government need to forbid the distribution of illegal content, such as child pornography, and therefore they even go as far as to contract companies who can conduct poison index attacks to sabotage those illegal file sharings [9].

Index poisoning can also be used to disrupt botnet communications. A botnet [1, 3, 21, 7] is a network of infected computers (bots) running malicious software, such as trojan horses, worms and viruses. As shown in Figure 3, those bots are remotely controlled by an attacker (botmaster), and the botmaster therefore can exploit the botnet to initiate various malicious activities, such as email spam, distributed denial-of-service attacks, password cracking and key logging, with the cumulative bandwidth and computing capability of the botnet. Botnets use P2P file sharing systems as a tool to rapidly and effectively infect vulnerable peers with malicious bot programs (malware) disguised as regular files, and also the P2P overlay is employed to communicate with the botmaster. Since indexes are used in botnets to distribute the botmaster's commands, index poisoning is considered as one of the options to disrupt and mitigate the effects of botnets. Once the indexes used by botnets are detected, bogus data can be inserted preventing the bots from executing commands given by the botmaster. However, in the newer botnet implementations, index poisoning has its limitations, since index data is encrypted and commands are authenticated via a public-key cryptosystem [10, 21]. Each bot generates its own keys that are dynamically changed at a certain rate, which make it increasingly difficult to fight against a botnet.

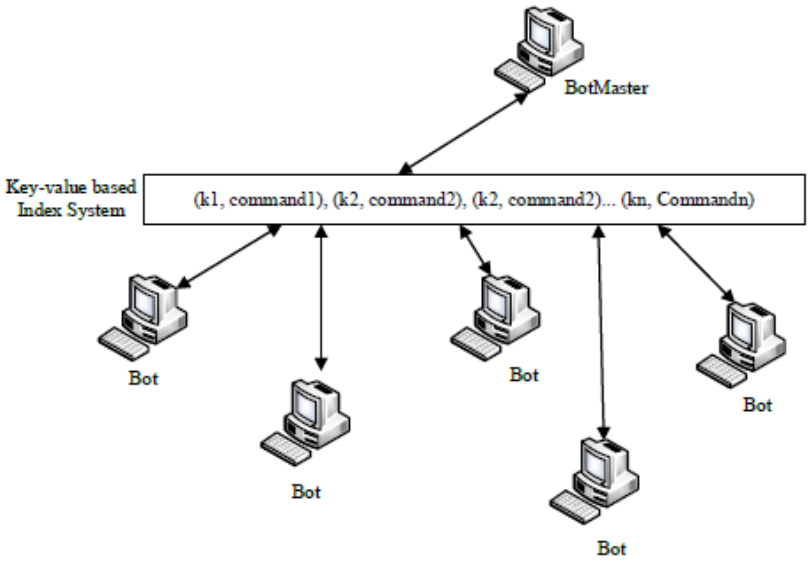

Figure 3 BotNet illustration

Yoshida et al. [23] utilizes index poisoning in Winny to prevent illegal file distribution, and to control the file distribution for anti-P2P companies. Winny is one of the most popular P2P file sharing networks in Japan. As an unstructured P2P network, Winny does not depend on any central server or super peers for file searching and sharing. A peer periodically send the KEY file of its sharing files to his neighbours, which includes the name, the identifier, the size, the location (IP address and service port) of the data file. Searching for a file is equivalent to finding the corresponding KEY, which is implemented by transmitting a file search query message. To control the illegal file distribution, the control peer diffuses a dummy KEY for the desired file, who has the same file identifier as the real one, but the file location is changed to a dummy peer. Note that the dummy peer does not provide any downloading service, instead it just sends a

handshake message to establish TCP connections with other peers. Since Winny protocol does not verify KEY files, a peer cannot distinguish between a genuine KEY and a dummy KEY.

Thus, when a peer search for his desired file, he always gets the dummy KEY, which eventually directs him to the dummy peer. As a result, control the distribution of illegal files is conducted.

Zhang et al. [24] propose PPBD, a piracy preventing system in BitTorrent (BT) Distributed Hash-Tables (DHT) networks, to stop pirated file sharing propagation without modifying its current architecture, nor affecting its legal users. The whole architecture consists of database, TCP session manager, DHT Sybil manager and some action engines. Database is used to store information of pirates, such as infohash and ip/port pairs of DHT peers. The TCP session manager handles incoming and outgoing connections and controls fake-block pollution process. The DHT Sybil manager handles incoming and outgoing UDP messages and controls poison and polluting processes over BT DHT. If a user wants to prevent the propagation of a pirated file, a torrent infohash can be submitted to the database via the user interface. Also, to prevent legal users from being blacklisted, the TCP session manager changes their IP/Ports periodically. Four peer sets are defined to implement the PPBD protocol: random peer set, critical peer set, index peer set, and tolerance peer set. Critical peer set is a set of peers whose routing tables contain most peer indexes of the target file. Index peer set includes peers that are peer indexes of the target file and are within the tolerance zone. In the index poison process, the crawler inserts the correspondent node into the Index Peer Set and inserts its previous node to the Critical Peer Set, which sends poisoned announce peer messages to critical peers and peer indexes. As a result, those peers in the DHT system assist to intercept announcement and querying message flows of real peer indexes pointing to source peers of pirate content. In this way, pirated file sharing propagation can be stopped with the index poisoning technology.

4. ATTACK PREVENTION

Regarding to the attacks caused by index poisoning, in this section, we investigate the existing prevention methods and classify them into two categories: Proactive and Reactive. Proactive methods act to block poisoning from taking place, while reactive methods minimize the impact or correct the poisoning of an index after poisoning has been introduced.

4.1. Proactive Methods

The proactive methods need to be conducted in advance in order to prevent any index poisoning attacks. The popular solutions include applying cryptography on information exchange, structure control, et al.

In [10], the authors propose the Reliable Index Exchange Protocol (RIEP) to address the threat of DDoS (distributed denial-of-service) attacks caused by file index poisoning into both of structured and unstructured P2P networks. This protocol utilizes the identity-based cryptography to establish peer accountability, while maintaining peers' anonymity. Peers' file indexes are traceable to the original publisher address without contacting the publisher. Specifically, RIEP uses two methods to insure accountability: identity bases signatures (IBS) and identity based cryptography (IBC). Those methods prevent attackers from pointing a poison index to a victim IP address. There are 6 steps involved in the protocol: private key request and assignment, index

publishing, index query, query forward, and index verification. The publisher requests a private key from the private key generator (PKG), and the request includes a nonce.The PKG responds with a private/public key pair generated along with a signature, and sends it to the publisher. In the index query phase the file index is constructed, encrypted and signed with the publisher's signature. A similar process takes place in query forwarding, where the query is signed by the peer who generated the query. A peer verifies the signatures of the indexes it receives with the publisher's identifier contained in the file index. Due to the peer accountability, it guarantees that each file index can be traced back to the publisher without user authentication. Also, attackers can only launch attack against themselves or the authorized IP addresses they used in the past, which therefore limits the attacker ability of choosing an external target.

In [11], authors improve their work in [10] by introducing an Accountable Indexing Protocol (AIP) for P2P file-sharing networks. AIP essentially proposes an IBS schema to protect P2P networks from modified and forged indexes by tracing any file index to the publishing peer. In a normal P2P network, the IP address contained in the index should be the same as the publisher's IP address since a peer should only advertise its own files. However, current P2P architecture does not verify this, so it is easy for a malicious peer to forge an IP address and port number causing DDoS attacks or connection errors. AIP address this specific issue by proposing changes to the file-indexing format and application layer protocol, where IBC is applied with a "4-tuple of doubly protected signatures". Using IBC over PKI significantly simplifies communication between peers, and allows the usage of a PKG over a Certification Authority.Furthermore, AIP implements a 5-step handshake. In the first two steps it interfaces between the PKG and the publisher through an encrypted connection to generate an AIP-index. The AIPindex will include the traditional index, AIP-encrypted key (combines the file index and publisher information), the certificate authority identifier and a digital signature. This AIP-index can then be distributed to the other peers. When a peer sends a download request, it first signs that request. The seeder receiving the request signs its reply (which contains the AIP information) with its own private key. The peer downloading can then simply verify the seeder's signature and publisher id for integrity. It is however recognized that the success of such a protocol relies on identities of the PKG being protected, which they have done by requiring each PKG to be register with a trusted certificate authority. It is interesting to notice that P2P architectures can also affect the vulnerability for the index poisoning.

In [9], authors investigates the poisoning attack in two file sharing systems, Overnet and astTrack, in order to develop an efficient methodology for estimating both index poisoning levels and pollution levels in generic P2P file-sharing systems. Overnet and FastTrack have very different architectural designs. Overnet is a DHT-based file-sharing system, which is an integral component of the popular eDonkey2000 file sharing system, and the index is distributed over all of the nodes in the system. On the other side, FastTrack is a two-tier unstructured filesharing system whose index is only distributed over super nodes, which is a relatively small fraction of the nodes in the network. To estimate poisoning and pollution levels, there are 3 different steps. First, it tracks the advertised copies over the measurement period, including the advertised titles and their publishers' information. Then, from those collected data, it determines which advertised versions are poisoned, which are polluted, and which are clean. Finally, it judges the poison and pollution levels, for both versions and copies. Authors discuss that structured overlays are known to be more efficient for searching, but all index values for a certain key word resides on one or a few computers, which creates a higher vulnerability. In a strictly unstructured overlay where each

node maintains a local index, an attacker must poison many nodes to create an impact. Therefore, choosing types of P2P architectures is one of the proactive option to resist index poisoning attacks.

Wang et al. [19] identify two routing table attacks caused by index poisoning, horizontal and vertical attack, and discussed their potential damages in BitTorrent Mainline DHT. Those two attacks are analyzed through honeypots in the real network. As a possible solution, authors propose that a node should explicitly check whether the ID in the PONG message is the same as the one in the routing table. Similar challenge should be introduced in the PING message. Such slight fix forces attackers maintain the state information for each previous contacted nodes, and also increases the computational overhead significantly, which can alleviate the horizontal attacks as a result.

4.2. Reactive Methods

The reactive methods of preventing index poisoning attacks include blacklisting, the reputation/voting scheme, the collaborative filtering and pollution modelling [12]. These methods are defined as reactive forms of prevention due to the fact that they rely on peers previous experiences to for perform any counteractive methods.

Liang et al. [9] suggest blacklisting as a means to avoid index poisoning attacks. Based on the proposed methodology, authors conclude that existing P2P file-sharing systems are highly vulnerable to index poisoning attacks. To defend against the index poisoning attacks, authors list two directions. One is to authenticate advertisements, and the other is to enable a user to rate sources from which they downloaded. The ratings are used to create a reputation for content publishers on the P2P network and determining whether a particular publisher is bad and should be blacklisted. One of the shortcomings of blacklisting is that newly joining peers do not have any reputation based on previous rating.

Costa et al. [2] proposes a hybrid peer and object reputation system to isolate active polluters, thus minimizing passive pollution dissemination and fighting pollution in P2P networks. The work is built on Scrubber (a peer reputation system) and Credence (an object reputation system). Authors observe that despite a quick convergence, Scrubber is not always able to clean polluted objects shared by peers that only occasionally upload them, and thus manage to keep good reputations. Credence, on the contrary, converges much more slowly, but is eventually able to isolate all polluted objects. By combining the benefits of both strategies, the proposed reputation converges much faster to a maximum efficiency, and is less sensitive to parameter setting, providing cost-effectiveness for various configurations. The issue with the reputation system heavily depends on the cooperation of reliable users. If a (even small) fraction of malicious users give the wrong feedback on whether an object is polluted, the performance suffers significantly.

Tauhiduzzaman et al. [20] investigate the reputation-based defence mechanisms against pollution attacks in P2P systems. In the reputation-based systems, each peer is rated by the other peers for reputation, and peers store their observations locally. Then they choose the data source for downloading based on the reputation ratings they have. Both of global reputation rating system, where peers periodically report their rating about other peers to a centralized server, and local reputation rating system, where peers exchange their ratings with each other to build the local

reputation rating table, are discussed. Also, it is interesting to note that the node degree of the polluters is a more dangerous indication than the upload bandwidth. When attackers are identified, exclusion is more useful to prevent attacks than limiting their connections. Various experiments to evaluate those two systems under different scenarios. Based on that, authors suggest that global reputation system does not necessarily improve the performance especially when collaborative attacks exist.

In [8], a PFtrust model which mixed trust-model based on reputation of peer with trust-model based on reputation of object is presented, to fight against pollution in P2P systems. In PFtrust model, each peer or file maintains a global reputation value which is kept in the super-nodes of the network. Node which wants to download a file firstly asks the system for a response list that contains the requested target files, their providers and reputation values. Finally, the peer chooses file from the list to download based on the reputation value of the file and the reputation value of the provider peer. When the downloading finishes, the peer reports the evaluation of the provider peer, and the evaluation of the downloaded file. Consequently, it identifies and isolates malicious attackers and contaminated files based on their global reputation. Punishment mechanism is also introduced to encourage peers to check and report the quality of files, and to control the spread of contaminated files actively and timely.

The Online Social Trust model (OST) [6] is another reactive prevention method intending to secure P2P exchanges and to prevent attacks like index poisoning, as well as to address internet security as a whole. OST is founded on the futuristic idea that social media and file sharing networks like P2P will one day form one. It attempts to solve the bigger picture of trust on the internet, which likewise also addresses P2P. OST proposes a trust model by which malicious activity is rated as untrustworthy and the P2P system will not allow that user to exchange information on that p2p network. This model would not depend on an authority, but rather monitor the users and adjust their rating based on their activities. The proponents of this model feel its solution can apply to both social networking issues and p2p. With this system in place users can define what level of trust is required for certain people to be part of an exchange system or community. The model is setup to make it easier to lose trust than gain it so there is in incentive to keep a good standing or be excluded by the system rules.

Shin et al. [17] propose to Winnowing, a hybrid (cooperative) approach to handle multiple types of pollution (index and file). Winnowing is designed to purify the index records (i.e. the information on files or the publishers) held by each index node in the system, so that download attempts based on these index records are more likely to yield satisfactory results. To achieve that, there are two steps involved. One is that index nodes performs preventive checks to block bogus publish messages upon receipt of a keyword or content publish message. The other is index nodes collect feedback from the users who have downloaded files via their index records, and the collected feedback is then processed and reflected in the matching index record. Specifically, files are added to the index through a publish message. An attack could be levied from outside the network if an attacker knows the proper IP addresses to send messages to. To prevent falsified information from being added by an outside source, a publisher verification message is sent to the publisher. Publishers that fail to respond are ignored and not allowed to publish content to the index. So an attacker must at least be a node of the system. An attacker would normally publish many keywords related to a targeted title for poisoning with faulty, or random hashes. When a node receives a keyword publish message, it verifies the message by issuing content search

messages, using as a target the content key in the keyword publish message. This greatly reduces the effectiveness of the attack.

In [14], authors present a method and the supporting architecture to detect and quantify the index poisoning pollution in the KAD network. KAD is a deployed P2P file sharing network implemented in eMule base on DHT. Each KAD node has a 128 bits ID determining its position in the DHT and the references of which it is in charge of. When sharing a file, the content and all the associated keywords are hashed separately to generate an ID which is then published into the DHT. The participating nodes use a double-indexation mechanism to index the publishing files. Firstly, the file's information are published towards the hash of each keyword (keywordID). Secondly, the peer publishes its own information (IP address, port, etc) towards the hash of the file to be indexed as a potential source. To detect the index pollution, authors use a modified aMule client in to collect, for a given file, all the file names advertised by the responding sources, and then evaluate their consistency.

4.3. Comparison

Both of the proactive methods and reactive methods can serve for unstructured and structured P2P networks. However, their performance are different when the pollution degree varies. Here the pollution degree means the number of index poisoning attacks in the network. If there is only a light weight index poisoning pollution in the network, the reactive methods are more efficient, since they consume relatively less resources. On the other side, the proactive always have to execute some schemes in advance, such as index encryption, no matter whether there are any index poisoning attacks. But when the pollution degree increases, which means large number of index poisoning attacks exist, proactive methods have better performance. That is because reactive methods have to consume more resources on collecting and processing each index poisoning attack. Such overload even increases when in the distributed reputation systems.

5. CONCLUSIONS

In this paper, we present a survey of the current index poisoning in P2P systems. Specifically, we summarize the usage of index poisoning, analyze the current prevention methods to resist the attacks caused by index poisoning, and divide them into two categories: proactive methods and reactive methods. We present the features of both these two categories, and make a comparison in the paper.

References

[1] S. Agarwal. "Performance analysis of peer-to-peer botnets using the storm botnet as an example". M.Sc. Thesis, 2010.

[2] C. Costa and J. Almeida. "Reputation systems for fighting pollution in peer-to-peer file sharing systems". In Proc. of Seventh IEEE International Conference on Peer-to-Peer Computing, 2007.

[3] C.R. Davis, J.M. Fernandez, S. Neville, and J. McHugh. "Sybil attacks as a mitigation strategy against the storm botnet". In Proc. of 3rd International Conference on Malicious and Unwanted Software, 2008.

[4] J. Kong, W. Cai, and L. Wang. "The evaluation of index poisoning in bittorrent". In Proc. of Second International Conference on Communication Software and Networks (ICCSN), 2010.

[5] J. Kong, W. Cai, L. Wang, and Q. Zhao. "A study of pollution on bittorrent". In Proc. of Second International Conference on Computer and Automation Engineering (ICCAE), 2010.
[6] M. Li, B. Alessio, and W. Zhou. "OST: A transaction based online social trust model for social network and file sharing security". In Proc. of IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC), 2010.
[7] X. Li, Y. Liu, and H. Zheng. "Peer-to-peer botnets: Analysis and defense". In Proc. of IEEE 3rd International Conference on Communication Software and Networks (ICCSN), 2011.
[8] Z. Li, J. Yao, and L. Shi. "The mechanism to control file-pollution based on hybrid trust-model in p2p network". In Proc. of IEEE International Conference on Anti-Counterfeiting, Security and Identification (ASID), 2011.
[9] J. Liang, N. Naoumov, and K.W. Ross. "The index poisoning attack in p2p file sharing systems".In Proc. of IEEE INFOCOM, 2006.
[10] X. Lou and K. Hwang. "Prevention of index poisoning ddos attacks in peer-to-peer file-sharing networks". In Multimedia, Special Issue on Content Storage and Delivery in P2P Networks,2006.
[11] X. Lou, K. Hwang, and Y. Hu. "Accountable file indexing against ddos attacks in peer-to-peer networks". In Proc. of IEEE GLOBECOM, 2009.
[12] J. Mao, Y. Cui, J. Huang, and J. Zhang. "Analysis of pollution disseminating model of p2p network". In Proc. of Second International Symposium on Intelligent Information Technology Application, 2008.
[13] X. Meng and W. Cui. "Research on the immune strategy for the polluted file propagation in structured p2p networks". Computers and Electrical Engineering, 38(2):194–205, March 2012.
[14] G. Montassier, T. Cholez, G. Doyen, R. Khatoun, I. Chrisment, and O. Festor. "Content pollution quantification in large p2p networks : A measurement study on kad". In Proc. of IEEE International Conference on Peer-to-Peer Computing (P2P), 2011.
[15] M. Ripeanu. "Peer-to-peer architecture case study: Gnutella network". In Proc. of 1st International Conference on Peer-to-Peer Computing, pages 99–100, Aug 2001.
[16] X. Shen, H. Yu, J. Buford, and M. Akon. Handbook of Peer-To-Peer Networking. Springer, 2010.
[17] K. Shin and D.S. Reeves. "Winnowing: Protecting p2p systems against pollution through cooperative index filtering". Journal of Network and Computer Applications, 35(1):72–84,January 2012.
[18] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H.Balakrishnan. "Chord: a scalable peer-to-peer lookup protocol for internet applications.Networking", In Proc. of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), 2001.
[19] L.Wang and J. Kangasharju. "Real-world sybil attacks in bittorrent mainline DHT". In Proc. of Global Communications Conference (GLOBECOM), 2012.
[20] M. Tauhiduzzaman; M.Wang. "A system analysis of reputation-base defences against pollution attacks in p2p streaming". In Proc. of IEEE International Conference on Performance Computing and Communications (IPCCC), 2012.
[21] P.Wang, L.Wu, B. Aslam, and C.C. Zou. "A systematic study on peer-to-peer botnets". In Proc. of IEEE Internatonal Conference on Computer Communications and Networks (ICCCN), 2009.
[22] M. Yoshida, S. Ohzahata, A. Nakao, and K. Kawashima. "Controlling file distribution in winny network through index poisoning". In Proc. of International Conference on Information Networking, 2009.
[23] M. Yoshida, S. Ohzahata, A. Nakao, and K. Kawashima. "Controlling file distribution in winny network through index poisoning". In Proc. of International Conference on Information Networking (ICOIN), 2009.
[24] H. Zhang, J. Shi, L. Ye, and X. Du. "PPBD: A piracy preventing system for BT DHT networks".In Proc. of INFOCOM, 2013.
[25] P. Zhang and B.E. Helvik. "Modeling and analysis of p2p content distribution under coordinated attack strategies". In Proc. of IEEE Consumer Communications and Networking Conference (CCNC), 2011.

Authors

Quan Yuan is an Assistant Professor of Computer Science at University of Texas of the Permian Basin. He received the Ph.D. degree of Computer Science from the Florida Atlantic University, USA. His research interest include wireless network,mnetwork security, mobile computing, and distributed systems.

Aaron Little is currently a Master student of Computer Science at University of Texas of the Permian Basin.

Maggie Kabore is currently a Master student of Computer Science at University of Texas of the Permian Basin.

Youssouf Kabore is currently a Master student of Computer Science at University of Texas of the Permian Basin.