Adding Trust to P2P Distribution of Paid Content

Alex Sherman[1], Angelos Stavrou[2], Jason Nieh[1], Angelos D. Keromytis[1], and Cliff Stein[1]

[1] Computer Science Department, Columbia University
[2] Computer Science Department, George Mason University

Abstract. While peer-to-peer (P2P) file-sharing is a powerful and cost-effective content distribution model, most paid-for digital-content providers (CPs) use direct download to deliver their content. CPs are hesitant to rely on a P2P distribution model because it introduces a number of security concerns including content pollution by malicious peers, and lack of enforcement of authorized downloads. Furthermore, because users communicate directly with one another, the users can easily form illegal file-sharing clusters to exchange copyrighted content. Such exchange could hurt the content providers' profits. We present a P2P system TP2P, where we introduce a notion of trusted auditors (TAs). TAs are P2P peers that police the system by covertly monitoring and taking measures against misbehaving peers. This policing allows TP2P to enable a stronger security model making P2P a viable alternative for the distribution of paid digital content. Through analysis and simulation, we show the effectiveness of even a small number of TAs at policing the system. In a system with as many as 60% of misbehaving users, even a small number of TAs can detect 99% of illegal cluster formation. We develop a simple economic model to show that even with such a large presence of malicious nodes, TP2P can improve CP's profits (which could translate to user savings) by 62% to 122%, even while assuming conservative estimates of content and bandwidth costs. We implemented TP2P as a layer on top of BitTorrent and demonstrated experimentally using PlanetLab that our system provides trusted P2P file sharing with negligible performance overhead.

1 Introduction

While P2P presents a powerful and cost-effective file-sharing model due to its ability to leverage the participating users' uplink bandwidth, most paid-content providers (CPs) typically rely on direct download methods to distribute their paid content. For example, Apple iTunes [14], Amazon [4] and Sony distribute content either directly from their website or via contracted content delivery networks (CDNs) such as Akamai [3]. The cost of content delivery, which involves either building infrastructure or paying CDN fees, is quite significant. While some CPs, such as Warner Bros. and AOL, have begun to experiment with limited P2P deployment [31] based on proprietary technology most CPs are reluctant to embrace the cheaper P2P content delivery model. Their worry is that unlike direct download, P2P introduces a number of security concerns such as unauthorized downloads of paid content and increased illegal content sharing that could reduce CPs' profits. In this paper, we introduce TP2P – an architecture that augments P2P to address these security concerns. We hope that our extension can help promote a

wider adoption of P2P by content providers and that the content delivery cost savings could benefit end-users via lowered content prices.

In a P2P model where peers download content from one another rather than from a centrally managed system, a number of security issues arise. These security threats include content pollution (as malicious peers could be serving garbage data to one another), unauthorized download of paid content (as it can not be enforced by a CP server or a CDN), and increased illegal file-sharing of copyrighted content by the P2P peers. Peers can protect against content pollution via standard hash-checking of the file chunks that they receive. However, in a P2P system peers do not have an incentive to enforce exclusively authorized downloads by other peers or to abstain from forming illegal file-sharing clusters with their neighbors. One reason that some CPs are hesitant to rely on a P2P model for content distribution is that it can easily deteriorate into a free file-sharing community similar to Xbox-sky [32] and Red Skunk Tracker [27]). Since P2P users communicate with one another directly during file distribution, it is easy for them to form clusters for an illegal file-sharing. To form a cluster, malicious users can use a simple protocol to "signal" one another as an invitation to join a cluster. Members of a formed cluster can exchange content that they have purchased or that they will purchase in the future from the CP. Thus they reduce CP's profits, as each member of the cluster only pays for a fraction of content that they obtain.

In order to protect against the threat of illegal file-sharing cluster formation and unauthorized downloads, we present a new technique that we call "trusted auditing". Trusted auditors (or TAs) are a new class of peers that police the P2P system by covertly monitoring other peers for any sign of misbehavior, such as admitting unauthorized users or protocol "signaling" that may lead to illegal sharing cluster formation. The TAs help detect and stop such cluster formation and thus protect the CPs profits. We model the behavior of TAs and show analytically and with simulations that TAs can effectively thwart the formation of illegal file-sharing clusters. By introducing this type of policing by trusted auditors we can provide security guarantees that are similar to those of direct download systems. We show via an economic model that since the cost of using the TAs is small TP2P can yield significant profits for the CPs.

When TAs detect misbehavior by a peer, a variety of countermeasures may be taken. For example, offending peers can be banned from the P2P system to a direct download system where they cannot exfiltrate any peer information. **We stress that we do not address illegal content sharing over out-of-band channels**. Sharing of files after they have been downloaded can happen regardless of the file distribution mechanism used by the CP: a user can download a movie via a CDN and then post it for free download on PirateBay [25]. However, it is important to address the threat of additional illegal sharing that may occur over the P2P delivery system used by the CP. Imagine the effects of having millions of iTunes users connected to a P2P system. Many of the users have demonstrated the willingness to purchase content. Regular users are reluctant to visit illegal pirated content sites because of the potential legal consequences as these sites are policed by RIAA, MPAA and third-party companies such as Media Defender [1]. At the same time, simple software can help iTunes users probe their P2P peers and invite them to share media libraries. Users know that their iTunes peers already have high quality purchased content and probably share similar interests since they have

learned one another's IPs by P2P sharing of the same content. With the use of TAs, TP2P prevents such additional illegal file-sharing when CP switches to the P2P model.

The contributions of our work are as follows:

• We introduce *automated trusted auditors* as a controlled and inexpensive way to monitor and detect certain types of misbehavior in a P2P system.

• We present an analytical model that shows how TAs effectively thwart malicious users from forming illegal file-sharing clusters. Our analysis shows that even where TAs are but a small fraction of all peers, they are sufficient at protecting the P2P system against unauthorized file sharing.

• Using a simple economic model we further show that TP2P provides a more cost-effective solution than direct download. This results in higher profits for a CP even in the presence of a large percentage of malicious users.

• Finally, we implement TP2P security elements on top of BitTorrent to demonstrate that our system can provide its functionality in an existing, widely-used P2P system with only modest modifications.

2 Related Work

As broadband Internet access becomes more prevalent, digital content stores such as Apple Itunes and Amazon have begun to distribute richer digital content over the Internet, such as TV series episodes and full-length movies. Since each download requires significant bandwidth, these stores typically contract Content Delivery Networks (CDNs) to distribute their content. Commercial CDNs include Akamai [3], Limelight [17] and VitalStream [30]. Since CDNs are centrally managed they can enforce appropriate security measures on behalf of a digital store, such as authorization of customers and encryption of served content. However, the price paid to CDNs for their services is quite high. Market research [2] suggests that digital media vendors spend 20% of their revenue on infrastructure costs for serving content. *While free academic alternative CDNs such as Coral [12] and CoDeeN [13] exist, these systems are typically limited in their deployment and the amount of bandwidth they are allowed to use.*

An alternative powerful distribution model is Peer-to-Peer (P2P) systems such as BitTorrent [6], Napster [21] and Kazaa [15]) among others. No extra contracted bandwidth is required as users leverage one another's upload links to "share" content. BitTorrent is perhaps the most popular of these systems, and many analytical works [34, 11, 16, 10] have shown the high efficiency and scalability characteristics of BitTorrent.

Some companies have begun to adopt the P2P model with some security measures. MoveDigital [20] implements a gateway in front of a P2P system to allow only authorized users access. However, once inside, users can leverage the system for further illegal sharing without limitations. For example, if a user can learn the IP addresses of other users inside the system, she can start sharing content with those users directly for free, bypassing the up-front payment. Moreover, users might choose to participate in the P2P system and pay to download files to gain knowledge about other participants that have similar interests. Then, they can easily form another, private P2P community, a darknet [5], for free future exchange of similar content

In contrast, TP2P is designed explicitly to guard against such free file sharing using an open system architecture that is resistant to exploitation even in the presence of malicious nodes. TAs used in TP2P are owned and managed by the content provider, and are unlike reputation-based systems [33] where users simply rate each other such that the resulting ratings may not be trustworthy.

An additional problem for efficient P2P distribution of content is "free-riding" by users who do not upload to their neighbors [19]. This problem can be partially addressed by BitTorrent's tit-for-tat mechanism [8] which was found to be fairly robust [18]. Additional solutions that consider incentives in P2P systems have also been proposed [29, 22, 24, 23]. We believe, that our technique of using TAs could also be used to solve this problem. We leave this idea as an item for future work and focus here on using TAs to prevent illegal cluster formation.

3 Architecture

The TP2P architecture is designed as an additional layer for common P2P systems. This layer consists of components that enforce stronger security and trust in the P2P system: the authenticator service and trusted auditors. While TP2P layer can be applied to virtually any common P2P system, we use BitTorrent as the underlying P2P system as a proof of concept. We selected BitTorrent given its popularity, open implementation, and its very efficient file-swarming mechanism where users share individual blocks of a given file.

The goal of BitTorrent is to distribute a file as fast as possible to all connected peers. BitTorrent splits the file (such as a digital movie) into a number of chunks. Participating *peers* exchange individual chunks of the file using a *file swarming* approach. The swarming algorithm is fully distributed and nodes use it to decide from which peers they are going to request their missing chunks. In addition, in each file-sharing instance there are one or more *Seeds* present. Seeds are peers that have all the chunks of the given file. The party that advertises the content typically initializes one or more *Seeds* with the full content of the file. A file-sharing instance also contains a *Tracker* that tracks all participating peers. A peer joins the system by contacting the Tracker. It receives a set of usually up to 50 IP addresses of other participating Peers. The Peer then exchanges chunks of the file with the other Peers and periodically updates its progress to the Tracker via *announce* messages.

3.1 System Overview and Usage

When the user decides to purchase content for the first time, she registers at the content provider's portal. She picks a username and a password and enters her payment information (*i.e.,* credit card number). She then downloads a software client that allows her to browse for files, purchase content and perform P2P downloads. For each purchase at the CP's portal she obtains a verifiable token (signed credential) that authorizes her to download the purchased content file. The authenticator also generates credentials for her to be used for secure communication during the download *session*. (We occasionally

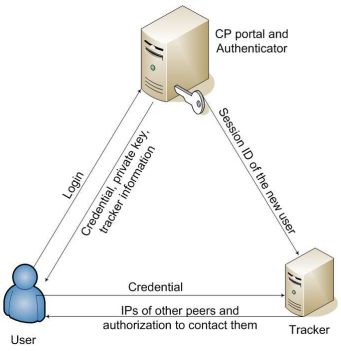

Fig. 1. To purchase a file, the user logs in on the portal, pays, obtains a signed credential and contacts the tracker for the purchased file.

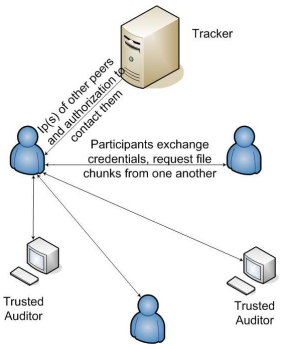

Fig. 2. Users authenticate one another and request file pieces. A fraction of trusted auditors is mixed in among the file-sharing peers.

refer to the file-sharing instance as a download session.) We describe these parameters in Section 3.2.

The user is then directed to a tracker that manages a file-sharing instance for the purchased file. The tracker validates that the user is authorized to perform the download by verifying her credentials. The user's interaction with the authenticator and the tracker is depicted in Figures 1 and 2. As in BitTorrent, the tracker *assigns* a set of other clients or *peers* to the new client. The client shares pieces of the purchased file with her assigned peers using BitTorrent's *file-swarming* approach. TP2P differs significantly from BitTorrent in the assignment of the peers. The TP2P tracker ensures that a certain fraction of the peers that it assigns are *trusted auditors* (TAs), as shown in Figure 2. TAs are special peers who, in addition to participating in a download session, detect misbehaving peers. The detected malicious peers are identified and "banished" from the system.

3.2 Authorization

We first describe the authenticator and other modules which enforce strong authorization and authentication. When a user purchases the content at the CP's portal, their credit card is charged the cost of the content. At that point the authenticator running on the CP's portal generates authorization credentials for the user (that will authorize her to participate in a download session for this content) and sends them to the user over a secure connection (using SSL). CP also stores the purchase record in case the user loses her credential due to reboot or another failure and needs to come back to the authenticator.

Authorization Credentials The authorization credentials given to the user include a temporary public/private key pair and a signed credential (akin to a public-key certificate) signed by the authenticator, whose public key is implicitly trusted by all participating users (*i.e.,* it is distributed along with the software, or is otherwise well known). More specifically, we use public-key-signed policy statements (similar in form

to public-key certificates [7]) issued by the content provider as the basis for authorization in our system. These credentials are given to authorized users after a purchase is made, and can be used as proof to both the Tracker and the other participants in a P2P download session. The credential includes a Session ID that identifies the user's download session, an expiration time, the user's IP address and public key, and an Instance ID (a unique identifier of the file-sharing instance managed by the Tracker).

Verification by Tracker Following the previous step, the user establishes an encrypted TCP connection to the Tracker using the Tracker's public key and sends its signed credential. The Tracker validates the digital signature of the credential against the authenticator's public key, confirms that the user's IP matches the one in the credential, and that the credential has not expired and that the Instance ID refers to a valid download instance.

If all the parameters are confirmed, the Tracker assigns and sends a list of other peers to the new user, along with a new credential that lets the new user contact other nodes of the same session.

Peer Verification When establishing communication, nodes that implement the correct protocol verify their peers using the tracker-issued credentials: the signature, IP address, public/private key binding, expiration, and instance ID. After verification, they negotiate a symmetric session key for their encrypted TCP connection using their public/private keys.

Certificate Revocation If a peer loses a credential, due to hardware down-time or local network down-time, she will re-login to the authenticator with her username and password and obtain new credentials. However, before the new credentials are issued the authenticator revokes the old credentials. The authenticator contacts the tracker previously assigned to this user and invalidates the old session ID. In response the tracker sends out new ACLs to the peers assigned to the peer with revoked session ID. The credentials and ACL revocation prevent a user from having multiple simultaneous identities in the system, thereby avoiding a scenario where a malicious user may attempt to steal and reuse the identity of an authorized user. Observe, that if the user machine is assigned a new IP by a local DHCP server after a network down-time, the new IP will be included with the new signed credentials.

3.3 Detecting Malicious Behavior with Trusted Auditors

The TAs imitate other peers through their participation in the P2P file exchange. In addition, they passively and actively detect malicious nodes that either allow download of unauthorized content or signal one another to form illegal file-sharing clusters. After malicious nodes are detected they are "banished" to an isolated direct-download system for future downloads. There they can no longer exploit the P2P system to form new content-sharing clusters. As a deterrent, the banished nodes may also be charged a penalty of the bandwidth cost for their future downloads. Alternatively, they may be warned with a temporary fine or threatened with legal action. The deterrent may vary according to the policy chosen by the CP as we discuss in Section 4.3. Since TAs also consume bandwidth and require a maintenance cost the relative number of the TAs must remain small.

Defining Malicious Behavior The system maintains and updates a definition list of malicious behavior (MDL) that can lead to unauthorized downloads and establishment of covert channels. The TAs can easily monitor the behavior described in the MDL. TAs can steer clear of false positives by following the full protocol that the malicious users use to exchange illegal content and incriminate them with evidence of such transaction. Initially the MDL includes unauthorized or unencrypted connections, connections to a non-protocol port and connections to a proper port that is not formatted according to TP2P protocol.

The CP employs two strategies in updating the MDL. One strategy involves actively searching, studying and running the software that malicious users use. The second strategy is learning the malicious probing format and pattern on the fly. This approach is based on recent work done at UC Berkeley on the RolePlayer system [9]. RolePlayer installed on a TA machine can quickly learn and replay various network communication patterns.

In order to form an illegal cluster malicious users attempt to discover one another by either establishing a covert channel or by accepting one. With high probability, the malicious node will probe a TA or reply to a probe from a TA and thus be detected and banished from the P2P system. Of course, here is also a small probability that a malicious node will find other malicious users by such probing and form a file-sharing *cluster* which diminishes with the size of the cluster. Thus, a more aggressive malicious node who may attempt to probe more neighbors aiming in forming a bigger malicious cluster, runs a higher risk of being detected by a TA and being banished to a direct download system. We explore and model the optimal strategy for a malicious node and the detection probability in detail in Section 4.

Behavior of Trusted Auditors TAs act as hidden "sentinels" in the system to prevent malicious probing, and therefore significantly limit the ability of illegal cluster formation. To stay hidden, TAs mimic different roles: regular or "neutral" nodes and malicious nodes. In their "neutral" role, TAs mimic the behavior of P2P peers by implementing the same discovery and download protocols, exhibit similar download speeds, arrival and departure rates as the regular clients. In their "malicious" role, TAs mimic the behavior of malicious nodes by sending out probes to their neighbors at the same rate as other malicious nodes.

3.4 Security Analysis

TP2P architecture was designed to ensure that a P2P content delivery system could exhibit similar security properties to a direct download system. In particular, we consider threats where users may attempt to exploit the P2P system by attempting illegal cluster formation and unauthorized downloads. We further classify the former threat into an *insider* and *outsider* attacks. In insider attacks a P2P participant may contact another participant during a download session. For outsider attacks, a node records the IP of it's peers and tries contact them from another IP address either during or after the download session. In the next session, we discuss how TP2P addresses those threats.

Insider Attacks One class of attacks against TP2P can stem from malicious users who purchase content and thus obtain the proper authorization to join a file-sharing instance. Such *insider* malicious users can then attempt to discover other malicious users among

the file-sharing peers and form a collaborative network for future unauthorized sharing. For instance, if five malicious users with similar interests discover each another during a file-sharing instance, then in the future only one out of five will need to purchase new content and share it with the rest. TP2P offers protection against this abuse by including TAs in the file-sharing network. The role of the TAs is to detect any malicious user attempting to scavenge information for future sharing. There are two ways in which TAs can detect malicious users: either because the malicious user contacts the TA and attempts to share unauthorized content, or because she allowed a TA to contact her and share content without proper authorization.

But how can we make sure that the identities of TAs are not exposed to the malicious nodes rendering them ineffective? There are two ways in which a TA can be exposed over time: either by learning the TA network locations (IP addresses) or by observing their behavior in the P2P system (*i.e.,* when they perform active probing or detect a malicious node). To avoid simple detection of the TAs' IP address pool based on their location, we can rent IP address space from Internet Service Providers based on their user population [28]. Moreover, for more sophisticated attacks that can learn even those IPs over time, we can request the TAs' IPs to be given via the same DHCP servers that the ISPs use for their own users. This will make the tracing of the TAs IPs futile since their IPs do not only change over time but are also shared with regular Internet users.

The second way to expose a TA is to learn to identify its behavior, in particular as it pretends to be malicious and probes other nodes. However, this is only true if malicious nodes already have the knowledge of what it is deemed a "normal" probing rate or they don't probe at all (thus exposing the TAs). In both cases, this requires some sort of previous shared knowledge among malicious nodes about the malicious behavior that they should exhibit. However even in the extreme case that malicious users have pre-agreed on a way of probing, the TAs can mimic such behavior because they are also receiving a fraction of the malicious probes. Thus, the TAs can adjust their behavior based on the probes that they themselves receive (remember that TAs communicate with one another their common knowledge about the received probing rates).

How can we protect the system from Denial of Service (DoS) attacks? Since TAs mimic the malicious node probing behavior, the increased rate of probing may cause TAs to amplify their probing and thus cause a DoS attack. To avoid this, we use randomized traffic thresholds for the probing rates received from the attackers. TAs do not probe beyond those rates. At the same time, malicious nodes that use DoS run the risk of being easily detected by the TAs. Thus, a DoS to scan for other malicious nodes in the P2P, even a short one, represent a prohibitive cost for the malicious user since the probability of being detected and shut down is high.

Outside Probing In this type of attack, an insider participates legitimately in a download session and collects the list of Peer IPs. It attempts to contact these IPs in search of other malicious nodes from an external IP either during or after the download session. Observe that contacts from outsiders who learn these IPs from a third party also fall under this type of attack. To address such outside scanning we use TAs who are not part of the P2P network to mimic the behavior of the outside scanning. Note that for a malicious node inside the P2P network there is little incentive to answer an outside probe. The reason for this is that outsiders are less likely to have content for trading. On

the other hand, nodes inside the P2P are far more likely to have content worth trading since they have proven that they are actually willing to buy such content. All things being equal in terms of scanning, by replying to outside probes malicious insiders run the same risk of detection with uncertain gains. In practice, there is no incentive for a malicious insider to respond to outside probes. The TAs prevent a possible DoS behavior by setting high random thresholds in the traffic they receive. Furthermore, as we show in our analysis in Section 4, the mere knowledge that TAs are present in the network causes rational malicious nodes to behave more cautiously and thus less dangerously towards the CP. TAs help to set the bar of malicious exploitation high by detecting malicious users who have purchased content and thus have gained authorized entry into the system. Furthermore, TAs detect users that do not honor (enforce) the authorization credentials generated by the authenticator.

Unauthorized Downloads Similarly, TAs probe the peers to check whether they allow unauthorized or unauthenticated downloads, by attempting to connect to them without proper credentials. Peers that deviate from the protocol by not enforcing the security checks are banned to the direct download system and may be selectively warned or penalized as a deterrent.

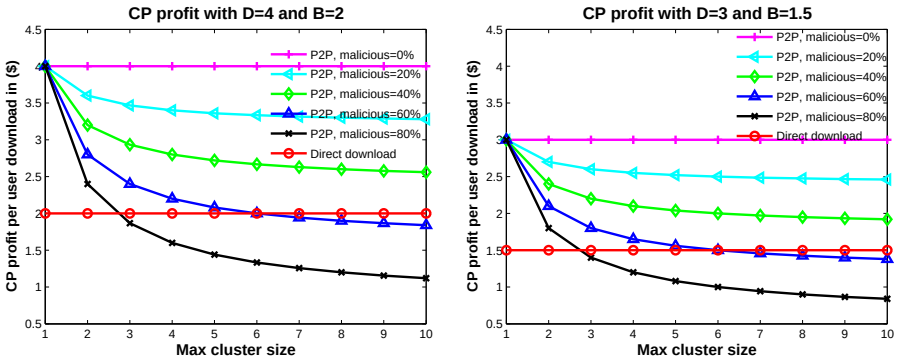

Fig. 3. CP profit per user download. Distinct combinations of D (profit before bandwidth) and B (bandwidth cost) capture variations in possible royalties and bandwidth agreements

4 Analysis

We have discussed two threats that may exist in the P2P framework for paid content distribution: unauthorized downloads and illegal file-sharing cluster formation. In the case of the first threat the user that allows an unauthorized download has nothing to gain by deviating from the TP2P protocol. In the case of the second threat each user that joins a cluster gets some content for free from the other cluster members. In this section, we focus our analysis on the threat of cluster formation and effectiveness of TAs in thwarting such activity. We analyze the strategy of malicious nodes and show that even a small number of the TAs can effectively curb the growth of clusters and successfully protect the CP's profits.

4.1 Economic Impact

We propose a simple economic model to quantify the impact that malicious nodes have on the CP's profit. We assume that the average price of digital content sold by the CP is S dollars. The CP pays a large part of that price as royalties $\$R$ to the content owner (a movie studio for example), and retains $\$D$. ($D = S - R$). In a direct download system the CP also pays $\$B$ for the bandwidth required to serve a file of average size to the end user. Thus the CP's profit per movie purchase is, on average, $\$(D - B)$. The market research in [2] shows that digital movie and audio stores pay roughly $60 - 70\%$ of end price (S) in royalties and the cost of bandwidth amounts to about 20%. Using a store similar to Apple Itunes as an example, one can purchase standard length (1GB) digital movies for \$10. We assume that D, the store's profit before bandwidth cost is \$3 to \$4 and B, the cost of bandwidth is roughly \$2 per download. We experiment with these assumptions in this section, but our results hold for wider ranges of values.

Using a P2P download approach the CP saves on most of the bandwidth cost and claims a full $\$D$ as profit. Unfortunately, in the presence of malicious users the CP collects smaller amount of revenue, and thus smaller profit since the malicious nodes form clusters to avoid full content payment. For example, if two malicious users manage to discover each other in the P2P system they will form a cluster of size 2. Then, these users will take turns purchasing files and sharing them with each other for free instead of buying them through the CP. For simplicity, we assume that malicious and non-malicious (or *neutral*) users desire to accumulate files at the same rate (e.g. say they download one movie per week), and that their interests are similar and thus they only need to purchase files at a fraction of the rate of the neutral users. For instance, in a cluster of two malicious nodes they each purchase movies at half the rate of the neutral. In general, users who belong to a cluster of size K need to purchase content at a $\frac{1}{K}$ fraction of the rate of the neutral users to get the same number of files in a given time interval. This scenario is pessimistic, since we assume that we lose from all malicious clusters whereas in practice, only some of the users in the cluster will want any particular file.

A single download session consists of up to N_s nodes that are all assigned to one another by a tracker. For a popular file, the system runs multiple download sessions of up to N_s nodes each. We assume that a single session contains at most M malicious nodes, T TAs and Q neutral nodes with $N_s = Q+M+T$. In a BitTorrent network a typical value of N_s is around $50 - 60$ nodes, thus in our system we will assume a maximum bound of $N_s = 100$. Let M_i be the number of users in the system who are malicious and who belong to clusters of size i. Then $M = \sum_{i=1} M_i$. We define $m_i = M_i/(M + Q)$ and $m = M/(M + Q)$ to denote the ratio of malicious users to the total number of malicious and neutral nodes. The amortized profit received by the CP for each file is:

$$\text{Profit} = D \cdot (1 - m) + D \cdot \sum_{i \geq 1} \frac{m_i}{i}. \tag{1}$$

The first term in Equation 1 is the CP profit from neutral users who pay a full price. The second term is from malicious users who pay only a fraction $\frac{1}{i}$ of the price based on their cluster size i (assuming multiple downloads). On the other hand, the profit of the

CP in a direct-download system per download is $D-B$. As a reminder, we do not attempt to solve *out-of-band* sharing that can exist with both direct and P2P systems. Rather, we are interested in curbing file-sharing from clusters formed by malicious exploitation of the P2P distribution system itself.

Using Equation 1, we produce the CP profit plots varying D and B. Figure 3 depicts CP profit curves for the P2P and the direct download systems for values of m ranging from 0 to 80%. Each plot uses different values for D and B to allow for variations in the cost of royalties and bandwidth. The x-axis shows the maximum size of a malicious cluster, K. The y-axis shows the average profit claimed by the CP user download. Each plot contains two horizontal lines: the top one representing a profit of a P2P system assuming no malicious nodes and the bottom one representing profit of a direct download system. The difference between the two plots is exactly B, the cost of bandwidth per download. The non-linear curves plot Equation 1 and represent the profit of a P2P system with various fractions m of malicious users. The plots show that as the fraction of malicious nodes and the file-sharing clusters that they form grow the profits for the P2P system dwindle. In fact, as the malicious nodes' fraction approaches 80% and for malicious clusters of > 20 nodes, the CP collects less than half the profits of a direct download approach. Even for less aggressive collections of malicious users, we see that most of the economic advantage of P2P rapidly diminishes.

4.2 Probing Game

We model the interaction between the malicious nodes and TAs as a *probing game*: malicious nodes probe and reply to probes from other malicious nodes in order to form and grow a malicious cluster. To detect malicious nodes, TAs also pretend to be malicious. They actively send probes and reply to probes of malicious nodes. To avoid being detected a malicious node must not probe all of its neighbors. Instead, she chooses a finite strategy that we call a growth factor (GF) which reflects the minimum cluster size that she aims to belong to at the end of the download session. The malicious node probes and replies to probes until she discovers at least $GF - 1$ other malicious nodes which may include a TA pretending to be malicious. For $GF = 1$ malicious nodes behave as a neutral nodes. If $GF > M$ the malicious nodes are certain to hit a TA and thus become detected before they can grow into a cluster of size GF. Thus, $1 < GF \leq M$.

In general, we make the following assumptions: malicious nodes remain "active" (i.e. they send and reply to probes) until they reach their growth factor of GF. Each malicious node knows both M and T in a download session, and based on that picks the most profitable value of GF. We suggest a good value for GF later in the section based on a simulation of multiple games. In the end of the session if a cluster formed during the session includes a TA (that pretended to be malicious) all the malicious nodes in the cluster are assumed to be "detected" and they are warned and "banished" by the CP. Such nodes still do not know which of the cluster nodes was trusted and thus cannot assume that they can share with the nodes they already discovered. Both malicious nodes and TAs send probes to randomly chosen neighbors at the same probing rate per node. TAs send probes at the same rate to be indistinguishable from malicious nodes. Otherwise, collaborating malicious nodes could easily pick out TAs in the system.Upon receiving probes, neutral nodes simply ignore them.

Figure 4 shows the probability that a malicious node *succeeds* in forming the desired cluster size. Here the fraction of malicious nodes in the download session m is fixed at 50% and the number of trusted T is varied over different ratios of M/T. The x-axis shows the strategy (i.e. growth factor) chosen by the malicious nodes in the game. The y-axis gives the probability that a node succeeds in achieving reaching its selected growth factor. As an example, the scenario of $M/T = 1$ (the number of malicious nodes and TAs is the same) and a target $GF = 2$, shows that the probability of a node succeeding in forming a cluster of size 2 is about 25%. Thus there is a $3/4$ chance that a node gets detected in such a game. An important observation about this plot is that all curves are decreasing monotonically. That means that as the malicious nodes become more aggressive by picking larger growth factors, they are also more likely to be *detected*. Interestingly, even for the top curve ($M/T = 10$) and the least aggressive target of $GF = 2$, there is only a 77% chance that such a node succeeds (*i.e.,* there is a 23% chance that it becomes detected). So, even in a favorable scenario, the probability that the node does not become detected in k independent games is roughly only $.77^k$.

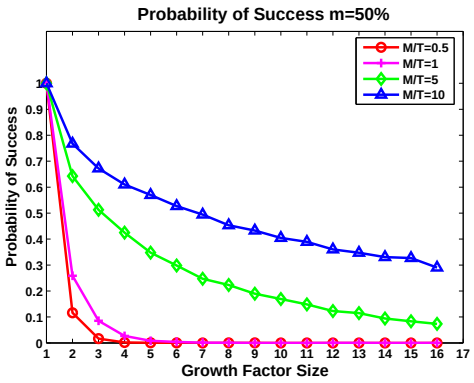

Fig. 4. For a single game, probability that a malicious node succeeds in forming a cluster of at least its growth factor with 50% of malicious users

4.3 Simulation and Results

We used MatLab to simulate the overall behavior of a BitTorrent-like P2P system with neutral, malicious and trusted nodes. We varied the overall system size, ranging from 10^5 to 10^7 participants. Our results remain consistent for all sizes. The plots presented in this paper are obtained using a population of $2 \cdot 10^6$ nodes. Our aim was to examine the performance limits of our system under diverse operating conditions by varying both the fraction of the malicious nodes M and their relative ratio to the TAs M/T. In addition, we wanted to find which growth factor is more beneficial for the malicious nodes across multiple downloads. We picked 30 downloads as the number we use for the multiple plots, because at 30 downloads we have detected the overwhelming majority of the formed clusters for all M/T ratios we consider. In addition, after 30 downloads, we notice that new clusters are formed almost exclusively by the new malicious arrivals and thus we consider the distribution to be stable. We assumed a *renewal rate* (departure and arrival of new users between downloads) of 5%. (Higher renewal rates result in even less effective cluster formation for malicious nodes).

In Figure 5, we present results from multiple downloads and for growth factors $GF = 2$ and $GF = 3$. The depicted results indicate that there is very little difference in the malicious cluster size distribution (CDF) when comparing $GF = 2$ and $GF = 3$ with the first having slightly better results. Therefore, the malicious users should select

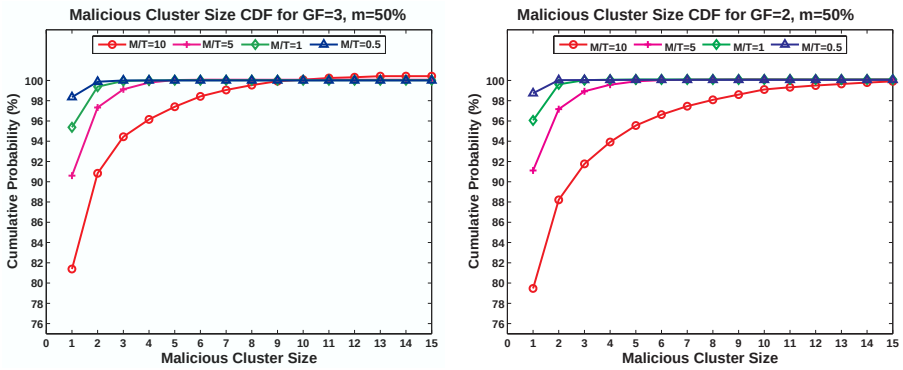

Fig. 5. Cumulative probability of forming clusters for growth factor $GF = 2$ and $GF = 3$ for multiple games. Notice that both plots look similar and that for $M/T = 10$, $GF = 2$ results in slightly larger cluster sizes.

$GF = 2$ as their growth factor if they want to optimize their probability of being in a larger cluster over multiple downloads.

The main result of the system with TAs is a high detection rate of the malicious nodes. In fact, in our simulation even starting with $m = 60\%$ of malicious nodes and $M/T = 10$ with $GF = 2$ after the multi-game simulation reaches steady state we observed that more than 99% of the malicious nodes in the system have been "detected". 80% of the malicious nodes failed to form clusters of even a small size prior to detection.

With the *conservative* policy, the CP warns the detected malicious users but leaves them in the P2P system. The CP threatens a fine or court action for illegal activity and forces them to re-download a new software client. If the CP believes that almost all such users will behave neutrally then it continues to make $\$D$ in profit from these users. Equation 2 presents the amortized profit per download under this policy.

$$\text{AP} = D \cdot (1 - m) + (D \cdot \sum_{i \geq 1} \frac{m_i}{i}) - B \cdot \frac{T}{M + Q} \tag{2}$$

This is an extension of equation 1 with the additional term: $-B \cdot \frac{T}{M+Q}$ that accounts for the bandwidth used by the TAs normalized by the total number of malicious and neutral users in the system. Figure 6 compares the profits of an unprotected system with that of TP2P based on a multi-game simulation (with the parameters as describe above) when it reaches steady state. TP2P shows much higher profits. For instance with $m = 60\%$, $M/T = 10$, $D = 4$ and $B = 2$ the profit is 122% higher for TP2P. Observe, that if instead the CP decides to move the detected nodes to a direct download system and charge them a penalty of their bandwidth cost the equation 2 also describes the resulting profit. In this situation the CP still makes $\$D$ from each download.

With an *aggressive* policy the CP does not trust the detected users to behave neutrally after a warning. The CP moves the detected users to a direct download system but does not charge them a bandwidth penalty. These users are no longer a threat but the CP now loses $\$B$ of bandwidth cost for their downloads. Equation 3 shows the profit based

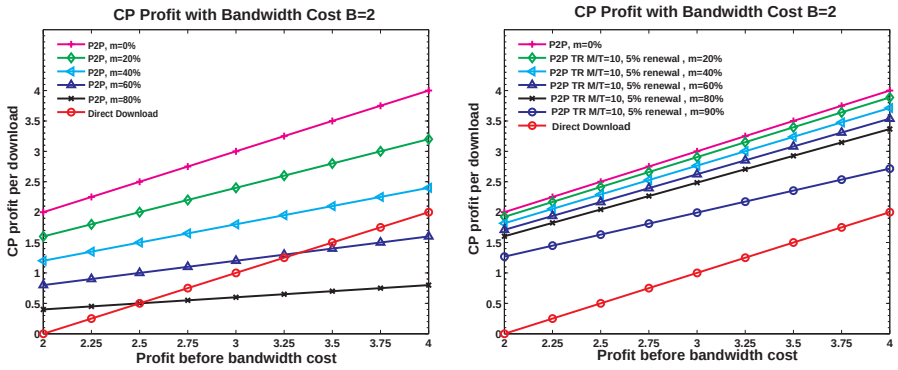

Fig. 6. Comparison of CP profits between a protected and an unprotected system for bandwidth cost of 2. On the left we have a system without TAs and on the right a system with a ratio of TAs to malicious users being 10. The protected system yields more profits that are comparable to a P2P system without malicious peers

on this policy where the CP loses $\$B$ on m_1 fraction of nodes - the singleton malicious nodes that are detected.

$$\text{AP} = D \cdot (1 - m) + (D \cdot \sum_{i \geq 2} \frac{m_i}{i}) + (D - B) \cdot m_1 - B \cdot \frac{T}{M + Q} \tag{3}$$

Even with this assumption in the case of $m = 60\%$, $M/T = 10$, $D = 4$ and $B = 2$ the steady state profit is 62% greater even with very few TAs. For a very high initial value of $m = 90\%$, the profit curve under this policy overlaps with direct download. The CP can improve the profit by moving detected nodes only temporarily until they can gain higher reputation. We leave this item for future study.

5 Implementation & Performance

We implemented an TP2P prototype by adding modifications to the existing BitTorrent client and Tracker (ver. 3.9.1) written in Python. Our modest modifications included adding secure channel communication using RC4 encryption, assignment of trusted auditors by the Tracker, and the distribution of credentials by the tracker to the peers. We conducted our experiments using PlanetLab [26] to compare the download speed of TP2P clients compared to BitTorrent clients on a set of geographically distributed machines given the overhead of secure communication and credentials distribution and verification in TP2P. Most machines used were equipped with 3GHz processors and ran the Linux 2.6.12 kernel.

For our first test, we deployed 41 BitTorrent clients randomly distributed in the continental US. A node was designated as the Seed client and initialized it with a 512MB movie file. To stress our system, we stored no parts of the file on the rest of the clients before the test. We ran the Tracker process on a machine outside of PlanetLab, a blade server with 3.06GHz processors, running a Linux 2.6.11 kernel, and a 10Mbit/sec upload bandwidth link. We ran the test both with the unmodified BitTorrent code and with

TP2P. The BitTorrent download times were only 0.8% faster on average, showing that TP2P adds negligible performance overhead.

For our second test, we performed a similar experiment as the first test but using a more dynamic scenario where peers join the download system at staggered times. We began with one Seed and 76 clients. The 76 clients joined the system at 2 minute intervals. By the time the later peers start, more clients in the system already have partial data sets. Therefore, newer clients have more sources to download the data from and thus their download times are generally faster. For this test, TP2P clients on average slightly outperformed BitTorrent by about 0.5%. This was due to the fact that the TP2P nodes contact the Tracker more frequently and receive new connection assignments at a faster rate at startup. As a result, initially they have slightly more choices for selecting faster sources. The CPU overhead on the TP2P clients was also minimal as RC4 encryption is a fast stream cipher. Average CPU utilization on the TP2P and BitTorrent clients was almost identical at roughly 1.3% and 1.23% respectively.

6 Conclusions

We introduced the concept of TAs to a P2P setting: by policing the system, TAs are able to enforce TP2P protocols and guarantee security properties that are similar to those of a direct download system. We have analyzed TP2P by modeling it as a game between malicious users who try to form free file sharing clusters and trusted auditors who curb the growth of such clusters. We have combined this analysis with a simple economic model to quantify the cost-effectiveness of our approach in the presence of malicious users. Our analysis shows that even when 60% of the participants in a system are malicious users, our system can detect 99% of malicious users and prevent them from forming large clusters, thereby providing strong protection of the P2P system against unauthorized file sharing. For most configurations, our analysis shows that TP2P yields profits that are between 62% and 122% higher than a direct download system based on conservative profit and bandwidth cost models. We demonstrate that TP2P can be implemented on top of BitTorrent with modest modifications, and provides its content protection and economic benefits with negligible performance overhead compared to vanilla BitTorrent.

7 Acknowledgements

This work was supported by NSF Grants CNS-07-14277, CNS-04-26623, and Google Inc.

References

1. About media defender. `http://rss.slashdot.org/~r/Slashdot/slashdot/~3/157024087/article.pl`.
2. J. G. Aguilar. personal communication, February 2006.
3. Akamai. `http://www.akamai.com/`.

4. Amazon. http://www.amazon.com/.
5. P. Biddle, P. England, M. Peinado, and B. Willman. The Darknet and the Future of Content Distribution. In *Proceedings of the 2nd ACM Workshop on Digital Rights Management*, November 2002.
6. Bittorrent. http://www.bittorrent.com.
7. CCITT. *X.509: The Directory Authentication Framework*. International Telecommunications Union, Geneva, 1989.
8. B. Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of P2P Systems*, 2003.
9. W. Cui, V. Paxson, N. Weaver, and R. H. Katz. Protocol-independent adaptive replay of application dialog. In *Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS)*, February 2006.
10. R. D. Qiu. Modeling and performance analysis of bittorrent-like peet-to-peer networks. In *SIGCOMM*, 2004.
11. R. D.Arthur. Analyzing the efficiency of bit-torrent and related peer-to-peer networks. In *SODA*, January 2006.
12. M. J. F. et al. Coral. http://www.coralcdn.org/.
13. V. P. et al. Codeen. http://codeen.cs.princeton.edu/.
14. Apple itunes. http://www.apple.com/itunes.
15. Kazaa. http://www.kazaa.com.
16. M. V. L. Massoulie. Coupon replication systems. In *SIGMETRICS*, 2005.
17. Limelight. http://www.limelightnetworks.com/.
18. N. Liogkas, R. Nelson, E. Kohler, and L. Zhang. Exploiting BitTorrent for Fun (but not Profit). In *International Workshop on P2P Systems*, 2006.
19. T. Locher, P. Moor, S. Schmid, and R. Wattenhofer. Free riding in bittorrent is cheap. In *5th Workshop on Hot Topics in Networks*, 2006.
20. Movedigital. http://www.movedigital.com/.
21. Napsterm. http://www.napster.com.
22. T. Ngan, A. Nandi, A. Singh, D. Wallach, and P. Druschel. Designing Incentives-Compatible Peer-to-Peer Systems. In *Proceedings of the 2nd Workshop on Future Directions in Distributed Computing*, 2004.
23. M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Building BitTyrant, a (more) strategic BitTorrent client. *USENIX ;login:*, 32(4):8–13, August 2007.
24. M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do Incentives Build Robustness in BitTorrent? In *Proceedings of the 3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2007.
25. Piratebay. http://thepiratebay.org/.
26. Planetlab. http://www.planetlab.org/.
27. Red Skunk Tracker. http://www.inkrecharge.com/ttrc2/.
28. Top 22 U.S. ISPs by Subscriber: Q3 2006. Market Research. http://www.isp-planet.com/research/rankings/usa.html.
29. V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. Karma: A virtual Currency for Peer-To-Peer Systems. In *ACM Workshop on the Economics of Peer-to-Peer Systems*, June 2003.
30. Vitalstream. http://www.vitalstream.com/.
31. Warner bros and p2p. http://arstechnica.com/news.ars/post/20060130-6080.html.
32. Xbox-sky. http://bt.xbox-sky.com/.
33. L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust in peer-to-peer communities. *IEEE TKDE, Special Issue on Peer-to-Peer Based Data Management*, 6, 2004.
34. G. d. V. X.Yang. Service capacity of peer to peer networks. In *INFOCOM*, 2004.