Can P2P Replace Direct Download for Content Distribution?

Abstract

While peer-to-peer (P2P) file-sharing is a powerful and cost-effective content distribution model, most paid-for digital-content providers (CPs) rely on direct download to deliver their content. CPs such as Apple iTunes that command a large base of paying users are hesitant to use a P2P model that could easily degrade their user base into yet another free file-sharing community.

We present TP2, a system that *makes P2P file sharing a viable delivery mechanism for paid digital content by providing the same security properties as the currently used direct-download model.* TP2 introduces the novel notion of trusted auditors (TAs) – P2P peers that are controlled by the system operator. TAs monitor the behavior of other peers and help detect and prevent formation of illegal file-sharing clusters among the CP's user base. TAs both complement and exploit the strong authentication and authorization mechanisms that are used in TP2 to control access to content. It is important to note that TP2 does not attempt to solve the out-of-band file-sharing or DRM problems, which also exist in the direct-download systems currently in use.

We analyze TP2 by modeling it as a novel game between misbehaving users who try to form unauthorized file-sharing clusters and TAs who curb the growth of such clusters. Our analysis shows that a small fraction of TAs is sufficient to protect the P2P system against unauthorized file sharing. In a system with as many as 60% of misbehaving users, even a small fraction of TAs can detect 99% of unauthorized cluster formation. We developed a simple economic model to show that even with such a large fraction of malicious nodes, TP2 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 TP2 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

The goal of our work is to make peer-to-peer (P2P) file sharing a viable delivery mechanism for paid digital content by providing the same security properties as the currently used direct-download model. P2P file sharing is a powerful and cost-effective content distribution model due to its ability to leverage the participating users' uplink bandwidth. Popular examples include BitTorrent [6], Napster [20] and Kazaa [15]). However, online paid-content providers (CPs) typically rely on direct download to distribute the paid content. For example, Apple iTunes [14], Amazon [3] and Sony distribute content either directly from their website or via contracted content delivery networks (CDNs) such as Akamai [2]. These CPs command a large base of paying users [18], and could directly benefit by using a P2P content distribution model, which would significantly reduce their infrastructure requirements and costs (or CDN fees), increasing profits and/or passing savings on to customers in the form of lower prices. However, CPs are hesitant to rely on a P2P model for content distribution, as it can easily deteriorate into a free file-sharing community similar to Xbox-sky [29] and Red Skunk Tracker [23]): since P2P users communicate with one another directly during file distribution, it is very easy for them to form clusters for free and illegal exchange of their past and future downloads from these CPs, *using the exact same protocols and, largely, software.*

If CPs are to successfully adopt a P2P distribution model, they need a system that does not allow their paying user base to deteriorate into yet another free file-sharing community, thereby eroding the CPs' profits. Designing such a system is challenging because in P2P systems, users communicate directly with one another and thus the user base is a fertile ground for proliferation of illegal file-sharing. First, the user base becomes quickly known to parties that want to promote unauthorized sharing. Second, the direct communication makes it easy for a few misbehaving users to distribute a modified or "malicious" software client quickly even to well-meaning users. The modified software automatically connects the users out-of-band and helps them share their libraries. *Out-of-band communication and distribution of content by misbehaving users (e.g., purchasing* *a movie on iTunes and making it available for free in Gnutella) is an equal threat for both the direct-download and P2P distribution cases, and our work is not directed against that scenario. We are motivated by the fact that, despite the wide availability of illegal file-sharing forums, commercially viable direct-download systems exist: Apple iTunes alone maintains tens of millions of paying users [18]. Our goal is to make P2P distribution as secure as the direct-download case.* Because P2P is an inherently more efficient system, we will reduce the costs for both the users and the CP.

We propose TP2, a system that meets the challenge of adopting the P2P model without the proliferation of unauthorized sharing. TP2 introduces the technique of "trusted auditing" that covertly monitors the behavior of P2P users and detects any sign of sharing-clusters formation, stops cluster-formation in its tracks, and thus protects the CP's profits.

TP2 allows a CP to take advantage of the cost-effective distribution capabilities of a decentralized P2P approach, while allowing the CP to maintain the same "trust" in its user behavior as with a direct download system. We view TP2 as a layer that adds security and monitoring capabilities to a generic underlying P2P distribution mechanism. First, TP2 adds strong authentication and authorization to a P2P system such that users can only download content after payment or other authorization. Second, it introduces the notion of *trusted auditors* (TAs), which are software agents nodes controlled by the CP that are posing as normal P2P nodes (users). Their purpose is to detect any type of unauthorized activity including deviations from the normal protocol implementation, port probing, malformed messages, attempts by unauthorized nodes to access content without proper credentials, and any effort to create an out-of-band communication with another node that is participating in the system. The detection of such forms of behavior is a key ingredient in preventing formation of out-of-band file-sharing clusters. When TAs detect misbehavior, a variety of countermeasures may be taken. In the simplest case, offending peers are banned from the P2P system to a direct download system where the CP does not need to worry about the peers exploiting the P2P file-sharing infrastructure.

TP2 provides the following key properties:

Content Protection. TP2 enforces strong authentication and authorization for all P2P communications, which inhibits unauthorized nodes from connecting to other compliant P2P participants. This property is enforced via encryption and management of signed authorization certificates.

Inherent Trust. TAs restrict the ability of malicious nodes to discover other malicious nodes for illegal file sharing and trading. As a result, the CP can have some guarantees that the P2P system will not be exploited by malicious nodes. Both with P2P and direct download, a user can share the content she downloads illegally in an out-of-band forum. However, with the use of strong authentication and TAs, TP2 protects the CP's user base itself from becoming a forum of free content downloads or information scavenging for future illegal trading. We demonstrate analytically how the probability of cluster formation in TP2 is significantly reduced.

Bandwidth Savings. Our system leverages P2P bandwidth for file downloads, providing tremendous scalable bandwidth and infrastructure savings. While the CP must provision bandwidth for the TAs, and therefore incur certain operational costs, we show that TAs constitute only a small fraction of the overall P2P network and thus the total bandwidth savings remain significant.

The contributions of our work are as follows:

- We introduce a novel technique of *trusted auditors* as a controlled and inexpensive way to automatically monitor and detect misbehavior in a P2P system.
- *We analyze TP2 by modeling it as a novel game between malicious users who try to form free file-sharing clusters and TAs who curb the growth of such clusters.* Our analysis shows that even a small fraction of TAs is sufficient to protect the P2P system against unauthorized file sharing. Even assuming that 60% of the P2P users in the system are attempting to form unauthorized clusters with other users, and 10 times as many misbehaving users as TAs, TP2 can detect 99% of the misbehaving users and can prevent 80% of such users from forming even small clusters. *Using a simple economic model, we further show that TP2 provides a more cost-effective content distribution solution, resulting in higher profits for a CP even in the presence of a large percentage of malicious users.* Even assuming that 60% of the P2P users in the system are attempting to form unauthorized clusters, TP2 achieves between 62 and 122% higher profit per download than a direct download system assuming conservative profit numbers and bandwidth costs.
- Finally, *we implement TP2 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.* Our experimental results on PlanetLab show that TP2 can provide its trusted content distribution functionality while imposing negligible performance overhead.

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 [2], Limelight [17] and VitalStream [28]. CDNs typically operate thousands of servers deployed in various networks and ISPs. In addition to offering vast amount of scalable bandwidth, CDNs 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 [1] 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 [20] 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 [31, 11, 16, 10] have shown the high efficiency and scalability characteristics of BitTorrent. Unfortunately, it is very difficult to implement proper authentication and authorization in such P2P systems to distribute copyrighted or paid content only to intended recipients. In fact, such systems typically implement a searchable directory of available content including copyrighted material. For this very reason, distributors of paid content shy away from P2P distribution models.

Various companies have attempted to address this problem with P2P systems. MoveDigital [19] 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.

Another approach is Avalanche [7], which uses network coding to encode exchanged blocks and relies on a proprietary protocol to attempt to prevent malicious use through security by obfuscation. However, if the system is hacked such that malicious nodes can participate in the system, there are no effective mechanisms to prevent its exploitation for free file sharing. In contrast, TP2 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 TP2 are owned and managed by the content provider, and are unlike reputation-based systems [30] 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 [24]. This problem can be partially addressed by BitTorrent's tit-for-tat mechanism [4] which was found to be fairly robust in [21]. Additional solutions that consider incentives in P2P systems have also been proposed [27][25]. We believe, that our technique of using TAs could also be used to solve this problem. By monitoring the fairness of user file-sharing behavior from multiple TAs the system can make a decision to move a user who is not contributing to its neighbors to a slower direct download system. We leave this idea as an item for future work and focus here on using TAs to prevent illegal cluster formation.

3. ARCHITECTURE

The TP2 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 TP2 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

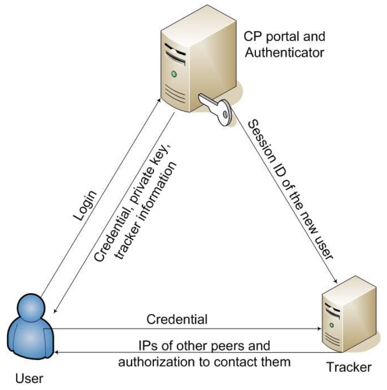
CP portal and Authenticator
Login
Credential, private key, tracker information
Session ID of the new user
Credential
IPs of other peers and authorization to contact them
User
Tracker

Figure 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.**

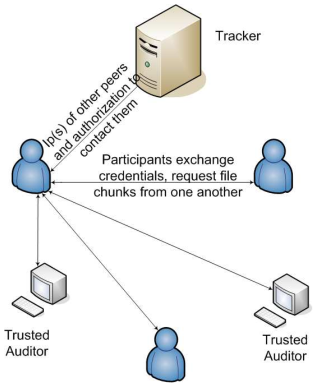
Tracker
Ip(s) of other peers and authorization to contact them
Participants exchange credentials, request file chunks from one another
Trusted Auditor
Trusted Auditor

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

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 subsequent purchase from the content provider (CP) at the CP's portal, she is authorized to perform the download by receiving a verifiable token (signed credential). The authenticator also generates credentials for the user to be used for secure communication during its download *session.* (We occasionally 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. TP2 differs significantly from BitTorrent in the assignment of the peers. The TP2 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 malicious communication. The detected malicious peers are identified and "banished" from the system as described in Section 3.3.

To summarize, the main distinctions between the TP2-augmented system and vanilla BitTorrent are:

- Whereas in BitTorrent a user contacts a tracker directly, in TP2 a user enters the system through the authenticator service. The authenticator hands out verifiable tokens and parameters for secure communication.
- The client and the tracker software is modified to enforce strong authentication, authorization checking and encryption in all point-to-point communications.
- In its assignments of peers in each download session, the TP2 tracker includes a fraction of TAs. The trusted auditors are owned and managed by the CP, and their identity is known to the tracker but not to the regular clients. The exact number of TAs is a parameter in the system that we discuss as part of our analysis in Section 4.
- TAs implement the same protocols as regular P2P clients, but also detect any malicious communication and "ban" malicious users from the P2P system as will be described further in the paper.

We now describe the new components added by TP2, and the threats that they help address.

3.2 Authorization

We now 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. An entry is also entered in the CP's database that authorizes the user with the given username to download the content. At that point the authenticator that runs on the CP's portal generates authorization credentials for the user and sends them to the user over a secure connection (using SSL).

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 [8]) issued by the content provider as the basis for authorization in our system.

These credentials are supplied 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. In total, 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 certificate. The Tracker validates the digital signature on the certificate against the authenticator's public key, confirms that the user's IP matches the one in the certificate, and that the certificate 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 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.

3.3 Trusted Auditors

In each download session, the system trackers include some TAs. The TAs imitate other peers through their participation in the P2P file exchange. In addition they passively and actively detect malicious nodes that either try to share content illegally or try to establish communication channels with other nodes for future illegal sharing. The discovered nodes are then banished to an isolated direct-download system where they can no longer discover other malicious nodes. As a deterrent, the banished nodes may also be charged a penalty of the bandwidth cost on their future downloads. Alternatively they may be warned with a temporary fine or threatened with legal action. The details depend on the policy of the CP and how they want to leverage the detection mechanism. We discuss some alternatives in Section 4.4.3. While TAs will reduce the loss (and thus increase profit) of the CP, they also incur a cost (*e.g.,* in terms of the bandwidth they consume, which the CP has to pay for). This is why the fraction of the TAs in a session must remain small.

Detecting Malicious Nodes As we describe in Section 3.4, malicious nodes may attempt to exploit the P2P system to either download content without proper authorization from the known IPs or to initiate a non-protocol connection with other IPs in the hope of forming an illegal cluster for future file-sharing. Strictly speaking, the TAs attempt to detect any communication that deviates from accepted TP2 protocols with the intent of establishing a covert trading channel. We can steer clear of false positives by following the full protocol that the malicious users use to exchange unpaid content and incriminate them with evidence of such transaction. Behavior that might lead to the establishment of covert channels include unauthorized or an unencrypted connections, connections to a non-protocol port and a connections to a proper port that is not formatted according to protocol. Again, all such activity can be monitored and reproduced by the TAs only when it leads to actual illegal exchange of content.

Malicious users can attempt to discover one another by either establishing a covert channel or by accepting one. If the malicious user is lucky, she will find a few other malicious users by such probing and can form a file-sharing *cluster* with them. However, if she is unlucky, she may probe a TA or reply to connection from a TA and thus be detected and banished from the P2P system. A more aggressive malicious node may attempt to probe more neighbors in hopes of forming a bigger malicious cluster, but at the same time it runs a higher risk of being detected by a TA and being banished to a direct download system. We explore and model the strategy of 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 excessive malicious probing, and reduce the rate of malicious cluster formation. To stay hidden, TAs mimic different roles: regular or "neutral" nodes, malicious nodes, and *seed* servers.

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. TAs employ two strategies: one strategy involves actively searching, studying and running the software that malicious users use on TAs. (We believe this to be a reasonable strategy, as the CP can invest significantly more resources than individual malicious users to obtain such software.) 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 can quickly learn and replay various network communication patterns.

Finally, TAs, in collaboration with the Tracker, serve in the capacity of BitTorrent *Seed* servers. This guarantees that all clients will be able to collect all the file pieces among their neighbors. To mimic users' behavior, TAs pretend that they join the system with no file pieces and gain more pieces over time.

Scalability of TAs The number of trackers and TAs should scale with the growth of system participants. As the number of users grows, so will the number of CP-owned machines. The cost of maintaining such machines should then scale with the growing revenue. The system does not require many physical TAs, as each such machine can participate in a number of simultaneous download sessions under virtual IPs. For instance, if the fraction of TAs in each file-sharing instance is 5%, and each machine participates under 10 virtual IP addresses, then for a 100,000 simultaneous participants only 500 physical TA-dedicated machines are necessary.

3.4 Security Analysis

The TP2 architecture was designed to ensure that a P2P content delivery system could be as equivalent as possible to a direct download system in terms of security and trust. This is challenging as in P2P systems users communicate directly with one another and user IPs become widely known over time. In practice, even with an obfuscated P2P client, a malicious user can easily extract the IPs of peers she is communicated with by running a simple system command. Wide knowledge of user IPs makes a P2P system open to both *unauthorized downloads* and *information ex-filtration* attacks which could lead to formation of user clusters for illegal file-sharing. Our aim is to develop methods that thwart both these types of attacks.

In our threat model, we assume that users have strong identities and use encrypted communication channels to avoid any leak of information from eavesdropping. In addition, we assume that any malicious users do not know one other a priori but rather attempt to find one other by participating in the system. This is a realistic assumption given the fact that we are trying to help CPs replace a direct download system with a P2P one, not to solve a generic DRM problem. If malicious users know one other using outside covert channels, then they can exploit the paid content distribution system regardless of the download mechanism.

These attacks pose a real and practical threat to a P2P distribution system. We discuss the two categories of attacks that are particularly aggravated by a P2P architecture: *content attacks* (*i.e.,* unauthorized download of content) and *information ex-filtration* (*i.e.,* attempts to form file-sharing clusters with known user IPs). We further classify the latter according to the logical position of the attacker: *Insider Attacks* (*i.e.,* attacks from system participants) and *Outsider Attacks.*

3.4.1 Content Attacks

In a P2P setting there is the risk of free downloading from unauthorized users who try to connect and extract content from the system. For example, in the BitTorrent protocol, if the IP addresses of the file-sharing peers becomes widely known, an unauthorized user can simply bypass the tracker, connect to BitTorrent clients running on those IPs and begin sharing pieces of the content with them. Another way to obtain content is to eavesdrop on unencrypted wired or wireless channels.

TP2 effectively prevents such exploitation by enforcing strong authentication and authorization over secure communication channels. The "Authenticator", module in TP2 is responsible for granting authorization credentials to users upon their entry into the system over an SSL-encrypted channel. Furthermore, after being authenticated, each user is granted a public key. The TP2 client individually enforces secure communication between each pair of authorized users using these public keys. Because a malicious client has to purchase content to gain a one-time system path, this creates a barrier for casual malicious users to enter the system to either obtain content or just to scavenge for IPs of other malicious nodes. Her unencrypted and unauthenticated requests are simply dropped by other participants.

3.4.2 Insider Attacks

Another class of attacks against TP2 can stem from malicious users who purchase content and thus obtain the proper authorization to join a file-sharing instance. As we discussed before, such *insider* malicious users can then attempt to discover other malicious users among the file-sharing clients and form a collaborative network for future unauthorized sharing. For instance, if five malicious users with similar interests discover one another in a file-sharing instance, then in the future only one out of five will need to purchase new content, that will then be shared with the group. TP2 offers protection against this abuse by including TAs in the file-sharing network. The role of the TAs is to detect a malicious user attempting to scavenge information for future sharing (*e.g.,*. 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 (impersonating a malicious user) to contact her and share content without proper authorization. Upon detection, the user is "banished" to an isolated direct download system as described in Section 3.3.

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 buy IP address space from Internet Service Providers based on their user population [26]. 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. By doing so, we make the tracing of the TAs IPs futile since their IPs do not only change over time but also they are shared by regular Internet users. Of course, our approach has to include IP address space inside universities and other large organizations which can be both from unused or DHCP based IP address pool. Obtaining that IP address space for the TAs does not have to be that extensive just representative. For example, for 500K simultaneous users, we require 25K (5% TAs). Given that 22 ISP have 75% of the total users in US [26] and the fact we only use the IP address space for tunneling packets, it is certainly feasible to obtain the necessary address pool.

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.

3.4.3 Outsider Attacks

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 as describe above. .

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.

4. ANALYSIS

In Section 3 we use strong authentication and authorization to prevent unauthorized downloads - the first threat discussed in section 3.4. Here, we are going to analyze the second threat - malicious nodes attempting to find one another and form file-sharing clusters. In these attacks, malicious nodes participate in the P2P network like regular users by registering and paying for movie downloads. If left unchecked, such malicious nodes may exploit the regular P2P system operation and extract peer information forming large malicious *clusters* for future illegal file-sharing. These clusters can exploit the system by purchasing only one copy of a file and sharing it among their members. As we will show with a simple economic analysis, such illegal sharing could significantly decrease the profits of the CP and be a strong deterrent against the adoption of a P2P approach for content distribution. In Section 3, we introduced TAs which limit the formation of large malicious clusters. In this section, we analyze further the malicious node strategies 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

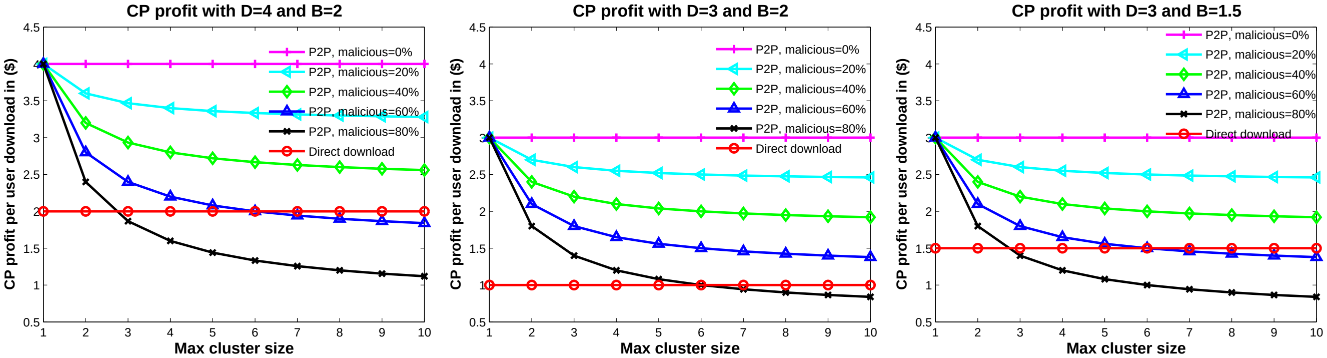

Figure 3: CP profit per user download. Distinct combinations of D (profit before bandwidth) and B (bandwidth cost) capture variations in possible royalties and bandwidth agreements

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 [1] 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 downloading 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. More generally, 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.

In our model, 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. We can now derive an amortized profit received by the CP each time a user accumulates a file as

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

The first term in Equation 1 is the CP profit from neutral users who pay a full price and 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 $D-B$. We remind the reader once again that 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 can produce the CP profit plots for various values of D and B. Figure 3 depicts CP profit curves for the P2P and the direct download systems for various values of m ranging from 0 to 80%. Each plot picks a different combination of values for D and B in a reasonable range as describe above 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 evaporates. If the malware that implements malicious code becomes readily available on the Internet, even the non-savvy users could easily become malicious. As the fraction of malicious users proliferates and they form larger clusters with time the profits of the CP quickly erode. For a P2P system to succeed, it is thus imperative to limit the effect of the malicious nodes. In the rest of the section, we show how even a small fraction of TAs could maintain the near-optimal P2P level profits for the CP.

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. Observe that for $GF = 1$ malicious nodes will not do any probing and will act exactly as a neutral node. On the other hand, 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, GF will take on some value in the range from 1 to M. In general, we make the following set of assumptions about a download session.

- Malicious nodes remain "active" (i.e. they send probes 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.

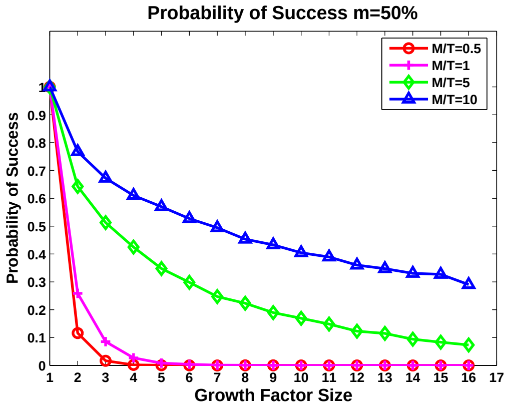

Figure 4: For a single game, probability that a malicious node succeeds in forming a cluster of at least its growth factor for m=50% (i.e. 50% of users are malicious)

- 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. Observe, that 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. (Having neutral nodes play a role in detecting malicious nodes can potentially help the detection of malicious but we leave it as an item for future work).

Our primary focus is to show that over time, as malicious nodes play multiple *games*, (i.e. they participate in many download sessions) most of them become *detected* and large clusters are unlikely to form. Small clusters may form, but these have limited economic impact. However, first we show some simulation-based plots to give some intuition about what happens in a single download session. We note that these plots use averaged probabilities collected from a thousand runs of a simulated game.

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 $.77^k$.

4.3 Analytical Model

We start by analyzing the single session download model and describing the growth of clusters both in the first session and after multiple sessions (downloads). To this end, we present a Markovian model with memoryless states and well-defined transition probabilities. Using this Markovian model we can fully compute the stationary probabilities of malicious cluster formulation (*i.e. the probability that a node ends up in a cluster of size K).* Moreover, our model can compute the stationary probabilities of the absorbing(final) states starting from any initial discrete cluster distribution. This enables us to compute the stationary distribution across multiple downloads by recursively applying our analysis starting with singleton malicious clusters and using the resulting cluster distribution as a starting distribution for the next session.

To simplify our presentation, we start by modeling a download session with a M and T malicious nodes and TAs respectively. Moreover, we set the value of the growth factor to be 2, ($GF = 2$). A state in such a download session is fully captured by the tuple $<A, F>$ where A is the number of active malicious nodes (that are still probing) and F the number of detected (found) nodes. The number of clusters of size two at a given state is simply the remaining nodes: $\frac{M-F-A}{2}$. The valid transitions from $<A, F>$ are to $<A-2, F>$ (two active malicious nodes form a cluster) and to $<A-1, F+1>$ (one active node becomes detected). Recall that all active malicious nodes and TAs probe at the same rate, and thus the probability that the next probe is from a malicious node is $\frac{A}{A+T}$ and is $\frac{T}{A+T}$ that it is from a TA. The probability of a cluster being formed is simply the probability that the next probe is sent by an active malicious node or that it is sent to one other active malicious node. (Recall that the probabilities do not need to account for probes being sent to inactive malicious or neutral nodes as such nodes simply ignore the probes.) We thus have the following transition probabilities for $F \geq 1$ and $A \geq 2$:

$$P_{<A,F>,<A-2,F>} = \frac{A}{T+A} \cdot \frac{A-1}{T+A-1}.$$

A node is detected when a malicious node sends a probe to a TA or when a TA sends a probe to a malicious node. For $A \geq 1$,

$$P_{<A,F>,<A-1,F+1>} = \frac{A}{T+A} \cdot \frac{T}{T+A-1} + \frac{T}{T+A}.$$

The starting state is $<M, 0>$ and the terminal states are of the form $<0, F>$ where no active nodes are left. The process is clearly Markovian as transition probabilities are uniquely determined by the current state, and the probabilities add up to 1.

We can generalize this models to the case with $GF > 2$. Here the state space becomes $<A_1, \ldots, A_{2\cdot GF-2}, F>$, where A_i is the number of clusters of size i. Note, that for a target size GF it is possible to have clusters of size up to $2 \cdot GF - 2$, because two active clusters of size $GF - 1$ could join together. Let $L = \sum_{k=1}^{GF-1} k \cdot A_k$ be the number of active nodes (i.e. all malicious nodes in clusters of size $< GF$). We can then proceed similarly to the case $GF = 2$ and compute the non-zero transition probabilities in three cases.

The first case is that a cluster of size i becomes detected, which occurs when a malicious node belonging to a cluster of size i probes a TA or a TA probes a malicious node in a cluster of size i.

$$\begin{aligned} &P_{<\ldots,A_i,\ldots,F>,<\ldots,A_i-1,\ldots,F+i>} \\ &\quad = \frac{i \cdot A_i}{T+L} \cdot \frac{T}{T+L-i} + \frac{T}{T+L} \cdot \frac{i \cdot A_i}{L} \end{aligned}$$

The second case is when two clusters of size i and j, with $i < GF$ and $j < GF$ join to form a cluster of size $i + j$. The third case is like the second but with $i = j$.

$$\begin{aligned} &P_{<\ldots,A_i,\ldots,A_j,\ldots,A_{i+j},\ldots>,<\ldots,A_i-1,\ldots,A_j-1,\ldots,A_{i+j}+1,\ldots>} \\ &\quad = \frac{i \cdot A_i}{T+L} \cdot \frac{j \cdot A_j}{T+L-i} + \frac{j \cdot A_j}{T+L} \cdot \frac{i \cdot A_i}{T+L-j} \\ &P_{<\ldots,A_i,\ldots,A_{2i},\ldots>,<\ldots,A_i-2,\ldots,A_{2i}+1,\ldots>} \\ &\quad = \frac{i \cdot A_i}{T+L} \cdot \frac{i \cdot (A_i-1)}{T+L-i} \end{aligned}$$

These are the only allowable and valid transitions. The starting state is $S =< M, 0, \ldots, 0 >$ means that all nodes are singletons (*i.e belong to clusters of size* 1). Moreover, all terminal states are of the form:

$$< 0, \ldots, 0, A_{GF} \ldots, A_{(2\cdot GF-2)}, F >$$

There are no clusters of size less that GF as all malicious nodes have either been detected or have formed clusters of size equal or more than GF.

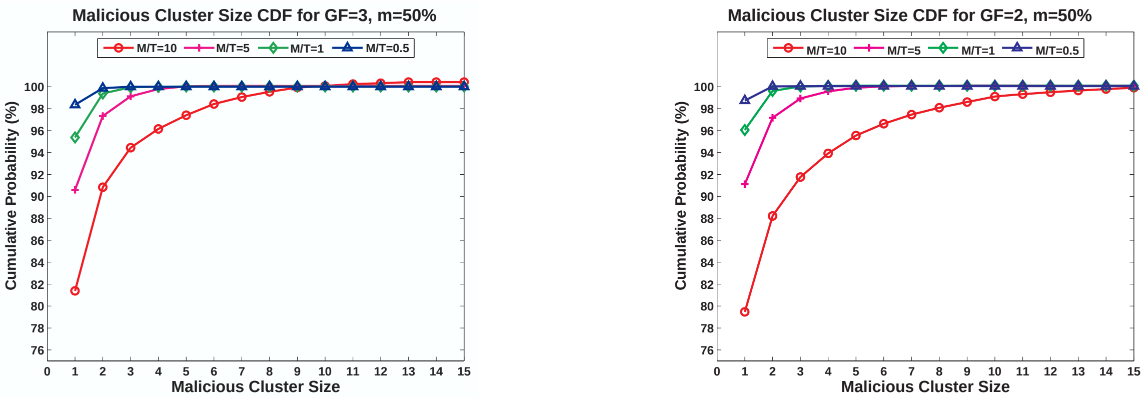

Figure 5: Cumulative probability of forming clusters for growth factor $GF = 2$ and $GF = 3$ for multiple games. Notice that both plots look similar and that for $M/T = 10$, $GF = 2$ results in slightly larger cluster sizes.

Once we construct the transition probabilities matrix P we can easily compute the probability of specific terminal states by exponentiation $P^{(n)}$ for some $n > 1$ at which the terminal probabilities converge. ${P_{S,j}}^{(n)}$ gives the probability that a particular terminal state j occurs. (Note also that n is bounded by $M * (GF - 1)$ as each malicious node transitions at most $GF - 1$ times).

4.4 Multiple Games (Downloads)

For multiple downloads, we have to consider that after the first download session, there are some undetected clusters formed. Furthermore, we assume that it is to the benefit of the malicious nodes to have only one representative from each cluster participate in the next download session since they share content out-of-band.(this way only one of them pays) For example, assuming a closed system (no new arrivals), if in the first download we had M malicious participants and all of them managed to form clusters of size 2 , in the second download only $M/2$ of them will participate. Therefore, after the first game, we will only have a fraction of the malicious nodes that participated in the first download either because they managed to form clusters or because they got detected by TAs. In addition, we assume that these malicious participants are acting on behalf of the formed clusters and thus are still aiming for the same growth factor GF.

The Markovian approach we presented in the previous section can still be applied but with some modifications: we have to recompute the transition probabilities because for each games we start with a different number of malicious nodes. We can compute the number of malicious nodes between games by taking into consideration the following parameters: formed clusters from previous game (both detected and undetected), arrival of new malicious singletons and departures of malicious nodes. Formed undetected clusters are the clusters who still contain nodes that have not been detected in previous downloads. In addition, we assume that we have Ar arrivals of malicious singletons and De departures of malicious nodes that can be either previously detected or undetected. For simplicity, we chose $Ar = De$ and we call the new quantity “renewal rate”. Throughout our analysis, we used a renewal rate of 5% but similar results hold for renewal rats of 1% to 10%.

Thus, based on the information on the clusters formed and detected from the previous game and the renewal rate, we compute the new number of malicious nodes that will participate in the next download. We then generate the transition probability matrix and we start from state $S =< M_n, 0, \ldots, 0 >$ where M_n is the number of computed malicious nodes. Using the resulting steady-state probabilities and the cluster distribution prior to the download, we compute the resulting joint distribution assuming that only one node from the undetected clusters participated in the download. For example, assuming that we have a representative of cluster of size K with F detected participating in a download with growth factor $GF = 2$ and initial cluster distribution P_i, then the resulting transitions are to a cluster of size $K + i$ with F detected with probability $P_s \cdot P_i$ and a cluster of size K with 1 detected with probability P_d, where P_s and P_d are the probabilities of actually forming the cluster or being detected, respectively.

Although, maintaining state for all formed clusters across multiple downloads appears to be large, in practice it is not. We start from a small initial set of malicious singletons that form clusters which grow by merging with each other, leading to rapidly declining number of non detected clusters. Moreover, the malicious to TAs M/T is also decreasing, leading to more detected malicious nodes. Therefore, although there are in principal a large number of possible cluster sizes that can be formed, in practice the actual number of formed clusters decreases fast over multiple downloads.

4.4.1 Simulations

To verify our analytical model and to avoid the com-

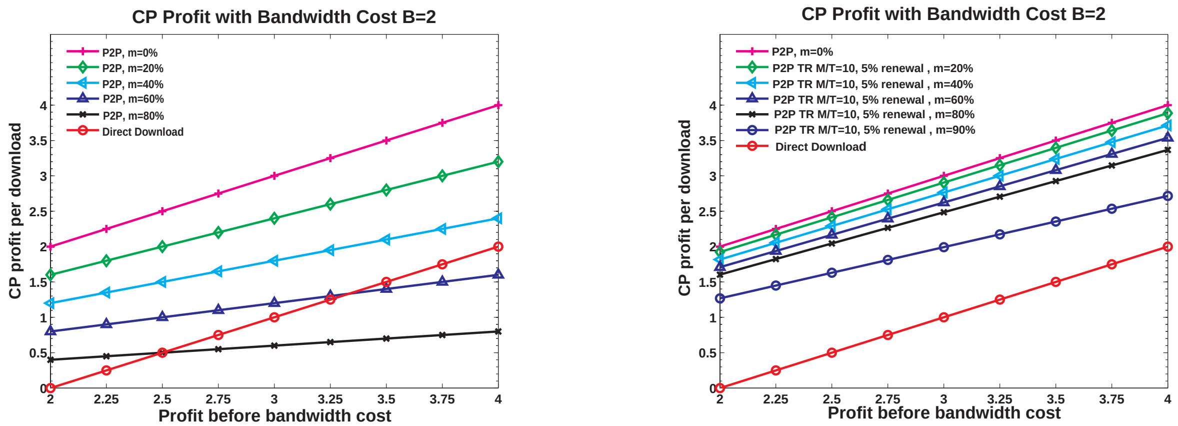

Figure 6: Comparison of CP profits between a protected and an unprotected system for bandwidth cost of 2. On the left we have a system without TAs and on the right a system with a ratio of TAs to malicious users being 10. Clearly, the protected system yields more profits than the unprotected system for the same fraction of malicious nodes which are very close to the ones produced by a P2P system with 0% of malicious nodes.

putational complexity involved in computing joint probabilities cluster sizes formed over multiple downloads of a large user population, we generated simulations of single and multiple download sessions. The results of our simulations fully agree with the ones obtained using the analytical model for growth factors $GF = 2$ and $GF = 3$. For larger values of GF, and to show that such choice of GF is not beneficial for the malicious nodes over multiple downloads, we relied on the simulations.

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.

4.4.2 Results

We now describe our concluding results about the system. We first study the affect of the parameter GF. 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 $GF = 2$ 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.

4.4.3 Profits

CPs can leverage the high detection rate of TP2 to recognize higher profits. The actual profit model will depend on the policy that the CP uses to deal with misbehaving nodes. We look at *aggressive* and *conservative* policies, each of which result in higher profits than in a unprotected system.

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.

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

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 TP2 based on a multi-game simulation (with the parameters as describe above) when it reaches steady state. TP2 shows much higher profits. For instance with $m = 60\%$, $M/T = 10$, $D = 4$ and $B = 2$ the profit is 122% higher for TP2. 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 describes the profit based on this policy where the CP looses $\$B$ on m_1 fraction of nodes - the singleton malicious nodes that are detected.

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

Figure 3 shows the steady state profit based on such a policy. Even with this conservative 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.

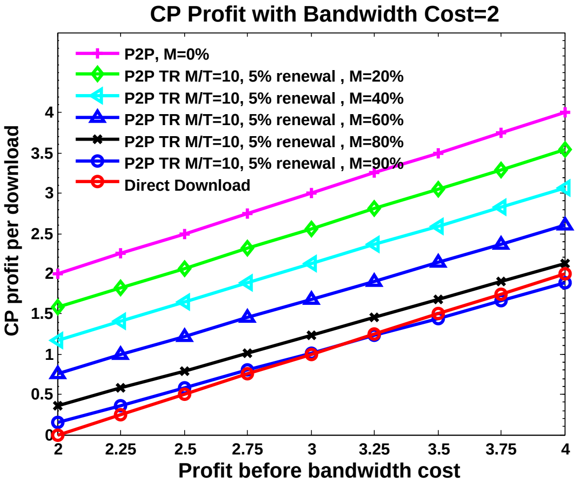

Figure 7: **CP profits in steady state based on a aggressive policy, with** $M/T = 10$

5. IMPLEMENTATION & PERFORMANCE

We implemented an TP2 prototype by adding modifications to the existing BitTorrent client and Tracker (version 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.

Due to space constraints, we briefly discuss two simple experiments we conducted using PlanetLab [22] to compare the download speed of TP2 clients compared to BitTorrent clients on a set of geographically distributed machines given the overhead of secure communication and credentials distribution and verification in TP2. 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 TP2. The BitTorrent download times were only 0.8% faster on average, showing that TP2 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, TP2 clients on average slightly outperformed BitTorrent by about 0.5%. This was due to the fact that the TP2 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.

6. CONCLUSIONS

TP2 is the first system that can be layered on top of existing P2P systems to enable content providers to leverage the download capabilities of a P2P system, and yet elevate their trust and content control to levels similar to that provided by a direct download system. Our approach provides strong authentication and introduces a novel notion of trusted auditors into a P2P system. Strong authentication ensures that the P2P system itself cannot be directly used for free file sharing. Trusted auditors appear as regular P2P nodes, but actively and passively detect malicious participants in the system to prevent scavenging of information that could be used to identify other participants for forming out-of-band free file sharing clusters. Just like in a direct-download system, TP2 does not prevent a user who legally downloads content from sharing it out-of-band, but it does prevent the system itself from being exploited in any way to facilitate out-of-band free file sharing.

We have analyzed TP2 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 TP2 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 TP2 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. We believe that our analysis and system provides a strong economic motivation for content providers to adopt regular P2P system enhanced with security guarantees for their content delivery. We hope that TP2 can serve as a strong foundation for creating practical P2P systems that can be used for paid content distribution.

7. REFERENCES

[1] J. G. Aguilar. personal communication, February 2006.

[2] Akamai. http://www.akamai.com/.

[3] Amazon. http://www.amazon.com/.

[4] B.Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of P2P Systems*, 2003.

[5] P. Biddle, P. England, M. Peinado, and B. Willman. The Darknet and the Future of Content Distribution. In *Proceedings of the* 2^{nd} *ACM Workshop on Digital Rights Management*, November 2002.

[6] Bittorrent. http://www.bittorrent.com.

[7] P. R. C. Gkantsidis, J. Miller. Anatomy of a p2p content distribution system with network coding. In *IPTPS*, February 2006.

[8] CCITT. *X.509: The Directory Authentication Framework*. International Telecommunications Union, Geneva, 1989.

[9] W. Cui, V. Paxson, N. Weaver, and R. H. Katz. Protocol-independent adaptive replay of application dialog. In *Proceedings of the* 13^{th} *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] Mac observer. http://www.macobserver.com/stockwatch/2005/10/17.1.shtml.

[19] Movedigital. http://www.movedigital.com/.

[20] Napsterm. http://www.napster.com.

[21] N.Liogkas, R.Nelson, E.Kohler, and L.Zhang. Exploiting bittorrent for fun (but not profit). In *International Workshop on P2P Systems*, 2006.

[22] Planetlab. http://www.planetlab.org/.

[23] Red Skunk Tracker. http://www.inkrecharge.com/ttrc2/.

[24] T.Locher, P.Moor, S.Schmid, and R.Wattenhofer. Free riding in bittorrent is cheap. In *5th Workshop on Hot Topics in Networks*, 2006.

[25] T.Ngan, A.Nandi, A.Singh, D.Wallach, and P.Druschel. Designing incentives-compatible peer-to-peer systems. In *2nd Workshop on Future Directions in Distributed Computing*, 2004.

[26] Top 22 U.S. ISPs by Subscriber: Q3 2006. Market Research. http://www.isp-planet.com/research/rankings/usa.html.

[27] 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.

[28] Vitalstream. http://www.vitalstream.com/.

[29] Xbox-sky. http://bt.xbox-sky.com/.

[30] 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.

[31] G. d. V. X.Yang. Service capacity of peer to peer networks. In *INFOCOM*, 2004.