A Measurement Study of Attacks on BitTorrent Leechers

Prithula Dhungel, Di Wu, Brad Schonhorst, Keith W. Ross
Polytechnic University, Brooklyn, NY, USA 11201
Email: pdhung01@students.poly.edu, dwu@poly.edu, brad@isis.poly.edu, ross@poly.edu

Abstract

Anti-P2P companies have begun to launch Internet attacks against BitTorrent swarms. In this paper, we analyze how successful these attacks are at impeding the distribution of targeted files. We present the results of both passive and active measurements. For our active measurements, we developed a crawler that contacts all the peers in any given swarm, determines whether the swarm is under attack, and identifies the attack peers in the swarm. We used the crawler to analyze 8 top box-office movies. Using passive measurements, we performed a detailed analysis of a recent album that is under attack.

1. Introduction

Over the past several years, the music industry has aggressively attempted to impede the distribution of copyrighted content over P2P file distribution networks. These attempts included numerous law suits against P2P file sharing companies (against Napster, Kazaa and many others), tracking and suing users of P2P file sharing systems [1], and most remarkably, launching large-scale Internet attacks against the P2P systems themselves. These large-scale Internet attacks were performed by specialized anti-P2P companies, working on the behalf of the RIAA and specific record labels. Several studies showed that these attacks were successful at severely impeding the distribution of targeted content over several P2P file sharing systems, including FastTrack/Kazaa, Overnet/eDonkey, and Gnutella [2, 3, 4]. These attacks, along with the law suits, have contributed to the demise of Kazaa and eDonkey file-sharing networks.

BitTorrent is one of the most popular P2P file distribution protocols today, particularly for the distribution of large files, such as high-definition movies, television series, record albums and open-source software distributions [5]. Unlike Napster and Kazaa, BitTorrent is nothing more than a protocol and about a dozen clients that implement the protocol. BitTorrent swarms and clients are not controlled by a small set of companies which can be targeted for a lawsuit. Also included in the BitTorrent eco-system are torrent location and tracker services, which can potentially be legally attacked; in fact, in late 2004, Suprnova, the largest torrent locater at that time, was closed after legal threats. Today, however, there are many BitTorrent file location and tracking services, scattered around the globe, all of which are defying legal threats, including PirateBay, Mininova, Snarf-it, and BiteNova. Moreover, torrent tracking can be decentralized using DHTs, as is currently being done with clients like Azureus and uTorrent.

Given that it is currently difficult, if not impossible, to stop BitTorrent by suing companies, and that suing individual users is both painstaking and unpopular, the only remaining way to stop BitTorrent is via Internet attacks. Not surprisingly, the music and film industries have begun to hire anti-P2P companies to impede specific "assets" from being distributed in BitTorrent swarms [6, 7].

In this paper, using Internet measurement, we explore how successful these anti-P2P companies currently are at impeding the distribution of targeted files in BitTorrent. We present results for both passive and active measurement. For active measurements, we developed a crawler that contacts all the peers in any given swarm, determines whether the swarm is under attack, and identifies the attack peers. We used the crawler to analyze 8 current top box-office movies. Using passive measurements, we performed a detailed analysis of a recent album that is under attack. For the passive measurements, we developed a customized packet parser, which identifies the peers that are attacking and the type of attack they employ.

2. Two BitTorrent Attacks

BitTorrent swarms are susceptible to a number of different attack types. In our measurement work, we have observed two attacks that are frequently deployed today, which we refer to as the *fake-block attack* and the *uncooperative-peer attack*. In this section, we describe these two attacks.

2.1. Fake-Block Attack

Recall that in BitTorrent, each file is divided into *pieces*, where each piece is typically 256 KBytes. Each piece is further divided into *blocks*, with typically 16 blocks in a piece. When downloading a piece, a client requests different blocks for the piece from different peers.

In the *fake-block attack*, the goal of the attacker is to prolong the download of a file at peers by wasting their download bandwidths. In particular, an attacker joins the swarm sharing the file by registering itself to the corresponding tracker. It then advertises that

it has a large number of pieces of the file. Upon receiving this information, a victim peer sends a request to the attack peer for a block. The attacker, instead of sending the authentic block, sends a fake one. After downloading all the blocks in the piece (from the attack peer and from other benevolent peers), the victim peer performs a hash check across the entire piece. The hash check then fails due to the fake block from the attacker. This requires the victim peer to download the entire piece (16 blocks) again, delaying the download of the file. If the peer chooses to download any of the blocks again from this or another fake-block attacker, the download is further delayed. Note that an attacker can cause a victim peer to waste 256 KBytes of download bandwidth by only sending it a 16 KByte block (using typical numbers).

2.2. Uncooperative-Peer Attack

In this class of attacks, the attacker joins the targeted swarm and establishes TCP connections with many victim peers. However, it never provides any blocks (authentic or fake) to its victim peers. A common version of this attack is the *chatty peer attack*. Here, the attack peer speaks the BitTorrent protocol with each of the victim peers, starting with the handshake message, and then followed by the bitmap message advertising that it has a number of pieces available for the file. When a victim peer requests one or more blocks, the attack peer doesn't upload the blocks. Moreover, the nature of the attacker is chatty. After the victim peer sends one or more block requests, the attacker resends the handshake and bitmap messages. By resending these BitTorrent control messages over and over again, the attacker persists as a neighbor, and the victim peer wastes a considerable time dealing with the attack peer, when it could have instead downloaded blocks from other benevolent peers. The effectiveness of this attack is increased if a significant fraction of victim's neighbors are uncooperative.

3. Effectiveness of BitTorrent Attacks

In this section, we use passive measurements to evaluate the effectiveness of fake-block and uncooperative-peer attacks on BitTorrent systems. In the next section, we complement this evaluation with active, crawler-based measurements.

3.1. Passive Measurement Methodology

While repeatedly downloading a file suspected to be under attack, we collected multiple packet traces from hosts connected to both Ethernet and DSL access networks. For this testing, we used *Azureus* and *uTorrent*, as they are the two most widely used BitTorrent clients. On each host, we ran Wireshark (or TCPdump) to capture all the incoming and outgoing packets. We also developed our own packet parser to identify different types of attackers in the trace and analyze their behaviors.

To measure the performance of BitTorrent without attacks, we use a third-party software, PeerGuardian [8], to prevent connections to and from the IP ranges in a specified blacklist. Our IP blacklist is based on the ZipTorrent blacklist published on *torrentfreak.com* [6]. Note that, since the anti-P2P companies (e.g., MediaDefender [9]) change the IP range of their attack hosts, this blacklist is not always complete and may not always eliminate all the attacker hosts.

3.2. Passive Measurement Results

In this section we present measurement results for a torrent for the new album titled *"Echoes, Silence, Patience & Grace"* from *"Foo Fighters"*, which we suspected to be under attack. This popular album was released on September 25, 2007, a few weeks before our experiments. At the time of the experiment, it held the number 1 position on the UK album chart and iTunes ranking list. The size of the file is 108MBytes. In our testing, we downloaded the file from this torrent 54 times.

3.3. Azureus Client

Because Azureus clients can import IP blacklist, we use this Azureus feature to perform IP filtering. Within one day, we performed downloads for this torrent multiple times using Azureus clients, and switched the IP filter on or off alternatively. First we present the basic average download-time statistics in Table 1.

Azureus	w/ IP-filtering	w/o IP-filtering	Delay Ratio
Ethernet	15.52 mins (6 downloads)	20.99 mins (6 downloads)	35.2%
DSL	19.98 mins (6 downloads)	25.88 mins (6 downloads)	29.5%

Table 1: Average downloading time using Azureus clients

In Table 1, *Delay Ratio* is defined as follows to evaluate the effectiveness of attacks in lengthening BitTorrent downloading time,

$$\text{Delay Ratio} = \frac{T_d \text{ w/o IP-filtering} - T_d \text{ w/ IP-filtering}}{T_d \text{ w/ IP-filtering}}$$

where T_d is the average downloading time of BitTorrent clients. From the table, we clearly observe that the downloading time of the file is prolonged when attacked. For both DSL and Ethernet peers, the download time on average increased by about 30%. The actual increase in download time may be larger, since we may not have blacklisted all the malicious peers. However, given the download rate of the DSL client, the size of the file, and that the minimum observed download time was 17 minutes, it is unlikely that the average download time without an attack would have been less than 17 minutes. Thus, we can safely say, at least for DSL, that the attackers did not prolong the downloading of this file by more than 50%.

To get a deeper understanding of the attack on Azureus clients, we selected one typical packet trace and analyzed it with the packet parser we developed.

Our parser can categorize all the IPs in the trace into different types as follows:

- *No-TCP-connection Peers:* peers with which our client cannot establish TCP connections.
- *No-BT-handshake Peers:* peers with which our client can successfully establish TCP connection, but when the client sends a BitTorrent handshake message, the peer does not return a BitTorrent handshake response.
- *Chatty Peers:* peers that just chat with our client. For Azureus clients, we consider any peer that sends more than one Azureus handshake message as a *Chatty Peer*.
- *Fake-Block-Attack Peers:* peers that upload fake blocks to our client. To identify fake-block-attack peers, we first need to check whether hash fails happened during downloading. When a hash fails, we identify all the IPs that have uploaded blocks for the piece and check whether the uploaded blocks are fake or not.
- *Benevolent Peers:* peers that communicate normally with our client via the BitTorrent protocol and upload at least one genuine block to our client.
- *Other Peers:* peers that don't fall into any of the above categories.

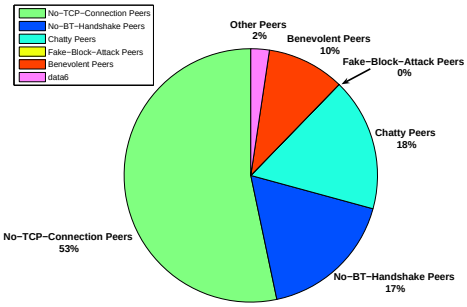

Figure 1: Peer distribution in Azureus trace

Figure 1 shows the distribution of different types of peers in the Azureus trace. Among all the IPs tried, the Azureus client could not establish TCP connections with over half of them. The high percentage of no-TCP-connection peers is not necessarily due to attackers. The no-TCP-connection peers include NATed peers, firewalled peers, stale IPs returned by trackers or gossiping messages, and peers that have reached their limit on TCP connections (typically around 50 in BitTorrent). Even in clean torrents (e.g., public-domain software) where no attacks exist, we observe a large percentage of no-TCP-connection peers.

No-BT-handshake peers account for 17% of the total IPs. If combined with no-TCP-connection peers, almost 70% of all the IPs are not useful for our Azureus client. For the remaining 30% of the IPs, only 10% of the IPs are benevolent peers, while 18% IPs belong to chatty peers, which chat with the Azureus client continuously but without any piece uploading. Chatty peers account for a majority of useful peers (i.e., 60%).

To estimate how chatty the attackers actually are, we checked how many handshake messages were sent out by each chatty peer. The results are shown in Figure 2. We can observe that most of chatty peers are very chatty, and send out as many as 40-60 handshake messages to our Azureus client. Those chatty peers persist as neighbors of the Azureus client during the downloading process, and hinder the client from contacting benevolent peers. No hash fails occurred during the downloading. Thus, it appears that the attackers did not launch a fake-block attack against Azureus clients at this time.

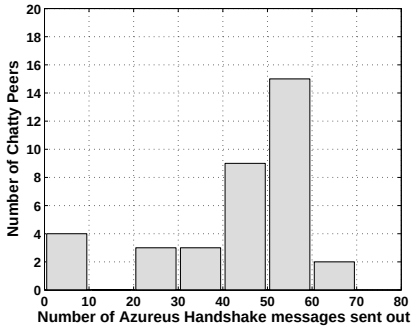

Figure 2: Distribution of Azureus handshake messages across chatty peers.

3.4. uTorrent Client

We also used uTorrent clients to download the same file. We turned off the automatic filtering function of uTorrent and used PeerGuardian to perform IP-filtering.

Table 2 provides the average downloading time of uTorrent clients. For uTorrent clients with Ethernet connections, the attackers did not succeed at significantly increasing the average download time. However, the attackers appear to have some success with DSL clients, increasing the average download time by 58%.

uTorrent	w/ IP-filtering	w/o IP-filtering	Delay Ratio
Ethernet	9.17 mins (10 downloads)	9.42 mins (10 downloads)	2.7%
DSL	18.32 mins (5 downloads)	28.93 mins (5 downloads)	57.9%

Table 2: Average downloading time for uTorrent clients

Table 3 shows the average number of hash fails for uTorrent clients. Compared with Azureus clients (which had no hash failures), hash failures occur much more frequently. The hash failures are a direct consequence of the fake-block attack being launched against uTorrent. Hash-failures may not significantly impact

uTorrent	w/ IP-filtering	w/o IP-filtering
Ethernet	1.7 Hash Fails	44.2 Hash Fails
DSL	4.2 Hash Fails	68.4 Hash Fails

Table 3: Average number of Hash Fails for uTorrent clients

an Ethernet peer, since if the Ethernet peer can find one other high-bandwidth benevolent trading partner, it may be able to rapidly download from it complete pieces (all 16 blocks) even if the other neighbors are producing hash failures. For DSL clients, because of the tit-for-tat algorithm, the client is typically trading only with other lower-bandwidth peers; even if one of these peers is producing a stream of clean pieces, the pieces would be coming in at a relatively low rate.

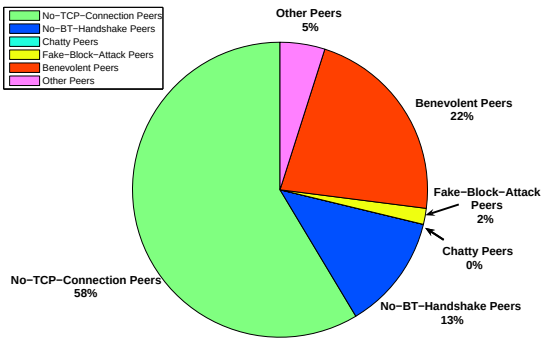

Figure 3: Peer distribution in uTorrent trace

To gain deeper insights, we plot the peer distribution in one of uTorrent traces in Figure 3. Similar to the Azureus trace, no-TCP-connection peers account for 58% of all the IP addresses in the peer list.

Compared with the Azureus trace, the main difference lies in the distribution of chatty peers and fake-block-attack peers. In Azureus, we saw significant chatty-peer activity but no fake-block attacker. In case of Azureus, the attackers exploited the implementation vulnerability of not being able to detect malicious behavior of attackers sending multiple handshake messages. It appears that uTorrent clients do not have this vulnerability.

However, we did observe the fake-block attack in uTorrent. The fake-block attack is different from chatty-peer attack in that it doesn't require many IP addresses to launch the attack. Even if the percentage of fake-block-attack peers is fairly low among all the IPs, the attack can still be effective, particularly towards the end of the file download (the end game).

In summary, the anti-P2P companies are applying distinctly different strategies against different BitTorrent clients. From this experiment (involving 54 downloads from the same torrent), we observe that the attacks are not always successful at significantly prolonging download times. For Ethernet clients, the attackers appear to be largely ineffective. On the other hand, average download times for our uTorrent client behind DSL connections increased by about 60%. However, even if download times were to double (100% delay ratio), it is not clear that many users would abandon BitTorrent, since users often download in the background or over night. In the next section, we examine the attack over a wider array of torrents.

4. Active Measurement Results for Top Box Office Movies

In this section, we provide active measurement results for the detection of chatty peers and fake-block-attack peers in torrents for 8 top box office movies.

4.1. Active Measurement Methodology

We developed a crawler that traverses the BitTorrent network gathering IP addresses of peers for a given torrent. We also developed customized BitTorrent clients, and devised heuristics for the detection of chatty peers and fake-block-attack peers. Doing this enabled us to quickly run experiments over a large number of torrents without having to download the entire files (as in the previous section).

4.1.1 Crawler Architecture

The output of the crawler is a "pool" containing the IP address and port pairs of peers in the torrent. It repeatedly requests the tracker for lists of peers participating in the torrent. Every time a list is received from the tracker, the crawler checks each IP and port to see if it already exists in the pool. If not, the new pair is added to the end of the pool. After gathering some IP addresses and ports in the pool, an IP address and port pair is extracted from the beginning of the pool. A separate thread is forked, which initiates a TCP connection to the peer.

If a TCP connection can be successfully established, the crawler thread then sends a BitTorrent handshake message to the peer indicating that it is an Azureus peer. If the peer is also an Azureus peer (which is determined from the handshake reply received from the peer), the thread speaks to the peer using the Azureus messaging protocol. An interesting feature of Azureus is that Azureus clients have the feature of exchanging gossip messages with each other for exchanging peer lists. Hence, by acting as a an Azureus peer, the crawler thread is able to gather additional IP addresses from the gossip messages that it gets from the Azureus peer. It is also possible to obtain peer lists by accessing a DHT created with Azureus clients, but we do not consider this feature in this study. The new pairs gathered via gossip are again added to the end of the pool.

A separate thread is forked for each IP and port pair in the pool and each thread runs until either there is error in the TCP connection with the peer, or the timer for the peer expires. Similarly, the whole crawling process is continued until the timer for the crawler expires. We tested our crawler on a number of torrents

Table 4: Measurement Results for Chatty Peers

Movie ID	Total Peers Crawled		IPs from Blacklist		Non-Useful Peers		Useful Peers	Chatty Peers		
	Tracker	Gossip	Tracker	Gossip	Tracker	Gossip		Tracker	Gossip	IPs from Blacklist
Movie1	116	864	1	73	90	836	54	0	27	26
Movie2	633	206	1	48	528	159	152	0	7	7
Movie3	144	158	0	30	111	98	93	0	0	0
Movie4	16	407	0	12	8	398	17	0	2	0
Movie5	29	1460	0	2	16	1460	13	11	0	0
Movie6	2356	3992	0	4	1992	3558	798	0	0	0
Movie7	111	0	0	0	81	0	30	0	0	0
Movie8	82	0	0	0	57	0	25	0	0	0

and observed that even for a swarm size as large as 12,000, the crawler was able to crawl more than 90% of peers within 8 minutes. In all of our experiments, we ran the crawler for 8 minutes.

4.1.2 Detection of Chatty Peer

For the detection of chatty peers, the instrumented client initiates TCP connections to IP addresses from the crawler pool. After having established a TCP connection, the instrumented client speaks the Azureus messaging protocol to the peer if the peer is an Azureus peer, and the "conventional" protocol in case of other peers. For peers that are Azureus clients, our client marks them as being "chatty" if they send more than one Azureus handshake message. Our client also marks a peer as "non-useful" if either a TCP connection cannot be made to it, or if it does not reply with a BitTorrent handshake message when a TCP connection is established.

4.1.3 Detection of Fake-Block Attack

For the detection of fake-block-attack peers, the instrumented client establishes TCP connections to peers from crawler pool and speaks the "conventional" BitTorrent protocol to all peers. In addition to marking peers as "non-useful," this client marks a peer as being "fake-block-attack peer" if the peer sends a fake block.

4.2. Active Measurement Results

We collected torrents for the 20 top box office movies during the time of the experiment. We ran an initial crawling on these torrents and checked the peer lists obtained against the blacklist. Out of the 20 movies, we chose the 3 movies (Movies 1 through 3) that appeared to be heavily attacked (based on the large number of blacklisted peers in the peer lists obtained from the crawler). We also selected 3 other movies (Movies 4 through 6) that appeared to be lightly attacked. For each of the 6 movies, we chose the torrent that showed the highest number of blacklisted peers for the movie. We also selected 2 other movies (Movies 7 and 8) that did not show any blacklisted IP addresses in their peer lists.

4.2.1 Test Results for Chatty-Peer Attack

Table 4 shows the test results for chatty-peer attack. We observe that for the 6 attacked movies, 70% (or more) of the peers crawled are not useful, meaning that they are either not reachable by TCP connections, or do not reply with a BitTorrent handshake message after a TCP connection is made. This result is consistent with our passive measurements in Section 3.

We also observe that for Movie 1, half of the useful peers (those who reply with the BitTorrent handshake message) are chatty. For Movie 5, about 85% of the useful peers are chatty. Interestingly, for Movie 5, none of the chatty peers that were detected fall into the blacklist that we have. As a verification, we downloaded the same movie using a real BitTorrent client and found that these IP addresses were indeed chatty. This indicates that static blacklisting is not sufficient for preventing such attacks since the attackers can always change IP addresses. Furthermore, for each movie, we observe a large number of blacklisted IP addresses from gossip. However, not all attack IPs come from gossiping - for Movie 5, there are 11 attacker IP addresses from the tracker and none from gossip.

For Movies 7 and 8, which did not appear to be under attack from the initial crawling, no chatty peers were detected and the percentage of non-useful peers is still around 70%.

4.2.2 Test Results for Fake-Block Attack

Table 5 shows the test results for fake-block attack for the same 8 torrents. It can be seen that the number of non-useful IP addresses is again 65% (or more) for the 8 torrents. For Movie 1, almost half of the useful peers were fake-block-attack peers. Since similar results were seen for the chatty peer test, it can be concluded that Movie 1 was indeed "heavily attacked" at the time of our experiments. Interestingly, at that time,

Table 5: Measurement Results for Fake-Block-Attack Peers

Movie ID	Total Peers Crawled		IPs from Blacklist		Non-Useful Peers		Useful Peers	Chatty Peers		
	Tracker	Gossip	Tracker	Gossip	Tracker	Gossip		Tracker	Gossip	IPs from Blacklist
Movie1	104	2284	6	68	75	2260	53	4	17	21
Movie2	604	313	1	72	494	255	168	0	8	8
Movie3	59	524	0	29	41	439	103	0	0	0
Movie4	15	86	0	10	10	77	14	0	0	0
Movie5	22	640	0	1	11	640	11	0	0	0
Movie6	374	884	1	1	292	677	289	0	0	0
Movie7	89	0	0	0	67	0	22	0	0	0
Movie8	114	0	0	0	74	0	40	0	0	0

it was already over 1.5 months after Movie 1 was released and so the movie was below 3 of the other 5 (attacked) movies in the box office rankings at that time.

We compared the list of chatty peers and fake-block-attack peers that were detected for Movies 1 and 2. We found that for each of these, some of the IP addresses detected as chatty were also detected as fake-block-attack peers. This reaffirms our claim that a specific attacker behaves differently for different BitTorrent clients.

In summary, our active measurement apparatus and methodology can quickly determine whether a torrent is under attack. We have found that several, but not all, top box-office movies are currently under attack. We have also found that published blacklists do not always cover all the attackers in a torrent. We also observed that the majority of the attack IPs enter the system through gossiping; however, some also enter through trackers.

5. Conclusion

We have seen that anti-P2P companies are currently launching Internet attacks to impede file distribution in BitTorrent swarms. We have identified two classes of attacks currently being employed: fake-block attack and uncooperative peer attack. We have found that these two attacks can indeed prolong the average download time of files, particularly for residential broadband users. However, the extent of this prolongation, at least for the torrents studied here, is modest - typically not by more than 50%. Since most BitTorrent users are fairly patient and download files overnight or in the background, we believe that download times need to be tripled (or at least doubled) to have a significant impact. Thus, the anti-P2P companies are not currently successful at stopping the distribution of targeted assets over BitTorrent. We have also found that blacklist-based IP filtering is insufficient to filter out all the attackers. To better filter out attackers, it is necessary to design smart online algorithms to identify different types of attackers.

Acknowledgment

We would like to thank Vishal Misra for discussions at the early stage of this research.

References

[1] A. Banerjee, M. Faloutsos, and L. Bhuyan, "Is someone tracking P2P users?" in *Proc. of IFIP NETWORKING*, Atlanta, GA, 2007.

[2] J. Liang, R. Kumar, Y. Xi, and K. W. Ross, "Pollution in P2P file sharing systems," in *IEEE Infocom*, Miami, FL, USA, March 2005.

[3] N. Christin, A. S. Weigend, and J. Chuang, "Content availability, pollution and poisoning in file sharing peer-to-peer networks," in *EC '05: Proceedings of the 6th ACM conference on Electronic commerce*. New York, NY, USA: ACM, 2005, pp. 68–77.

[4] J. Liang, N. Naoumov, and K. Ross, "The index poisoning attack in P2P file-sharing systems," in *Proc. of IEEE Infocom*, Barcelona, 2006.

[5] A. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The BitTorrent P2P file-sharing system: measurements and analysis," in *The 4th Int'l Workshop on Peer-to-Peer Systems (IPTPS'05)*, Ithaca , NY, February 2005.

[6] "Ziptorrent blacklist," http://torrentfreak.com/ziptorrent-pollutes-and-slows-down-popular-torrents/.

[7] "Bittorrent servers under attack," http://news.zdnet.com/.

[8] "Peerguardian," http://phoenixlabs.org/pg2/.

[9] "Media defenders," http://www.mediadefender.com/.