Peer-to-Peer Netw Appl
DOI 10.1007/s12083-009-0046-6

On the feasibility of exploiting P2P systems to launch DDoS attacks

Xin Sun · Ruben Torres · Sanjay G. Rao

Received: 7 November 2008 / Accepted: 25 March 2009

Abstract We show that malicious nodes in a peer-to-peer (P2P) system may impact the external Internet environment, by causing large-scale distributed denial of service (DDoS) attacks on nodes not even part of the overlay system. This is in contrast to attacks that disrupt the normal functioning, and performance of the overlay system itself. We demonstrate the significance of the attacks in the context of mature and extensively deployed P2P systems with representative and contrasting membership management algorithms—Kad, a DHT-based file-sharing system, and ESM, a gossip-based video broadcasting system. We then present an evaluation study of three possible mitigation schemes and discuss their strength and weakness. These schemes include (i) preferring pull-based membership propagation over push-based; (ii) corroborating membership information through multiple sources; and (iii) bounding multiple references to the same network entity. We evaluate the schemes through both experiments on PlanetLab with real and synthetic traces, and measurement of the real deployments. Our results show the potential of the schemes in enhancing the DDoS resilience of P2P systems, and also reveal the weakness in the schemes and regimes where they may not be sufficient.

X. Sun (✉) · R. Torres · S. G. Rao
Purdue University, 465 Northwestern Avenue,
West Lafayette, IN 47907, USA
e-mail: sun19@purdue.edu

R. Torres
e-mail: rtorresg@purdue.edu

S. G. Rao
e-mail: sanjay@purdue.edu

Keywords Peer-to-Peer · Security · DDoS · Evaluation · Measurement

1 Introduction

Peer-to-peer (P2P) systems are rapidly maturing from being narrowly associated with copyright violations, to a technology that offers tremendous potential to deploy new services over the Internet. The recently released Windows Vista is equipped with its own, under-the-hood P2P networking system [3], and several commercial efforts are exploring the use of P2P systems for live media streaming [4, 15, 16]. Recent studies [9] indicate that over 60% of network traffic is dominated by P2P systems, and the emergence of these systems has drastically affected traffic usage and capacity engineering.

With the proliferation of P2P systems, it becomes critical to consider how they can be deployed in a safe, secure and robust manner, and understand their impact on an Internet environment already suffering from several security problems. P2P systems enable rapid deployment by moving functionality to end-systems. However, they are vulnerable to insider attacks coming from (potentially colluding) attackers that infiltrate the overlay or compromise member nodes.

Several works [8, 12, 32, 33, 38] have studied how malicious nodes in a P2P system may disrupt the normal functioning, and performance of the system itself. In this paper, however, we focus on attacks where malicious nodes in a P2P system may impact the **external** Internet environment, by causing large-scale distributed denial of service (DDoS) attacks on nodes not even part of the overlay system. In particular, an attacker could subvert membership management mechanisms,

Published online: 23 April 2009

and force a large fraction of nodes in the system to believe in the existence of, and communicate with a potentially arbitrary node in the Internet. Such attacks may be hard to detect as the packets arriving at a victim are not distinguishable from normal protocol packets. These attacks may be viewed as a particular kind of reflector attacks [27], however the scale and unique properties of P2P systems make them worthy of study in their own right. These attacks are in contrast to the traditional botnet-based DDoS attacks, where the attacker has control over a large number of machines by infecting them with a malicious program that takes instructions from her [17].

In this paper, as a first contribution, we show that a potential attacker can launch attacks of hundreds of megabits a second on an arbitrary Internet node such as a web server, by exploiting the popularly deployed file distribution system Kad [13], and the extensively deployed video broadcasting system ESM [10]. The systems represent contrasting applications, and involve different and representative membership management designs - structured DHT-based and unstructured gossip-based. Our attacks exploit fundamental design choices made by P2P system designers, and shed new insights on the interplay between membership management mechanisms, and the feasibility of exploiting P2P systems to cause DDoS attacks.

As a second contribution of the paper, we present an evaluation study of possible mitigation schemes, and discuss their strength and weakness. These schemes include (i) preferring pull-based membership propagation over push-based; (ii) corroborating membership information through multiple sources; and (iii) bounding multiple references to the same network entity. We evaluate the schemes through both experiments on the PlanetLab and measurement studies of the real deployments. Our results shed light on the potential of the schemes and in what scenarios and regimes they may be useful, and also point to the potential weakness in the schemes.

This paper extends our preliminary work which appeared in the workshop paper [39]. While [39] presented attack heuristics on Kad and ESM, in this paper we also present an evaluation study of several mitigation schemes (Section 4 to Section 6).

The rest of the paper is organized as follows. Section 2 presents vulnerabilities that we identified in the Kad and ESM systems. Section 3 shows results demonstrating the feasibility of exploiting these systems for DDoS attacks. Section 4 discusses the strength and weakness of several mitigation schemes. Section 5 describes the methodology we use to evaluate the schemes, and Section 6 presents the results. We discuss related work in Section 7, and finally conclude our paper in Section 8.

2 Vulnerabilities in P2P systems

In this paper, we focus on DDoS attacks triggered by exploiting the membership management algorithms of P2P systems. The membership management algorithms in a P2P system enable a node to join the group, and maintain information about other members, even though nodes may join or leave the system. To scale to large group sizes, typical nodes maintain knowledge of only a small subset of group members. While a large number of P2P systems have emerged in recent years, two of the most common approaches for membership management involve the use of distributed hash tables (DHTs) [24, 28, 29, 34, 42], and gossip-based algorithms. While popular file-distribution systems like BitTorrent [7] and eMule [13] originally relied on centralized servers (trackers) for group management, more recent versions use decentralized mechanisms based on DHTs. Many other systems such as ESM and CoolStreaming [10, 41] employ gossip-like mechanisms to maintain group membership information.

To demonstrate the generality of the issues discussed, we consider P2P systems targeted at applications with very contrasting properties and very different membership management designs. In particular, we consider file distribution and video broadcasting applications. Video broadcast applications are distinguished by their stringent real-time constraints requiring timely and continuously streaming delivery. Both applications are bandwidth-intensive, and large scale, corresponding to tens of thousands of users simultaneously participating in the application.

The particular systems we consider in this work include Kad[13] for file distribution and ESM [10] for video broadcasting. Kad is a DHT based on Kademlia [24], add is supported by the popular eMule [13] clients and other eMule-like clients such as aMule [5] and xMule [40]. To the extent of our knowledge, Kad is the largest DHT currently used, with more than one million concurrent nodes [35]. ESM is a video broadcasting system that employs gossip-based membership algorithms. It is one of the first operationally deployed systems and has seen significant real-world deployment [10]. We are motivated to use these systems given their extensive deployment, the contrasting applications they

represent, and the different yet representative membership algorithms they employ.

2.1 DHT-based file distribution:Kad

In Kad, nodes and files have IDs that are globally unique and randomly chosen from the same ID space of 128 bits. Each node maintains a routing table with a subset of peers that are part of the system. For any file F, there are "index nodes" which maintain a list of members that own a partial or complete copy of the file. These members are called "sources" of the file. Index nodes are not dedicated nodes but regular participants, who have an ID close to a file ID. For example, in Fig. 1a, node A wishes to download a file F. A must first discover I, the index-node for file F, and obtain a list of sources from I. To discover I, A will start by querying nodes that it has in its own routing table (i.e. nodes that it already knows). These nodes either are the index nodes for F, or can point A to nodes closer in the ID space to the file ID, which are likely to be index nodes. In our example, A will initially query B which is not an index node for file F. B responds with C whose ID is closer to the ID of F. Next, A queries C who responds with I. In general, this process can iterate several times, but given the properties of distributed hash tables, convergence of the search process is likely. In our case, I is the index node for F and will respond to A with a set of sources of F. Finally, A contacts each of the sources to begin the download process.

Kad performs a similar lookup process for keyword search, file and keyword publishing, and routing table maintenance. Note that all Kad control packets use UDP.

Vulnerability: Kad may be exploited to cause a DDoS attack on a victim that is not part of the Kad network by creating a redirection attack. Whenever the attacker node receives a query from a benign node, it returns a (fake) response that contains the IP and port of the victim node (indicating that the victim is part of the system) along with a fake ID for the victim which is closer to the target ID. This tricks the benign node into thinking that the victim node is part of the Kad system and is likely to be an index node. Hence the benign node may contact the victim for searches as well as other normal protocol operations. For example, Fig. 1b shows how a malicious node M can make a benign node A send a query to the victim V, which can be an arbitrary Internet host (may not participate in Kad). The attack can be magnified if many benign nodes query the malicious node when they conduct searches. In addition, a coalition of malicious nodes could further increase the magnitude of the attacks.

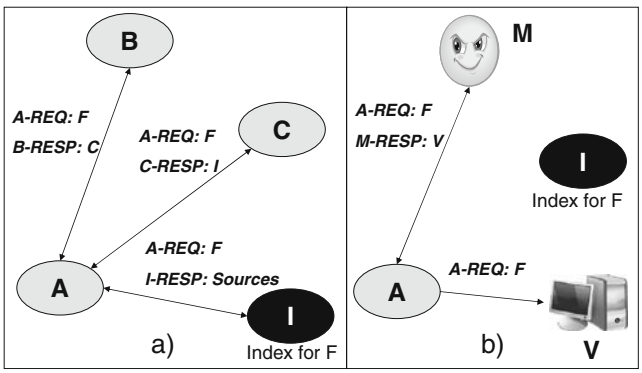


Fig. 1 **a**) Kad search mechanism. **b**) Redirection attack

2.2 Gossip-based video broadcast: ESM

ESM is a video broadcasting system built on top of an overlay network. It constructs a multicast tree for data delivery, which is primarily optimized for bandwidth and secondarily for delay. In addition, it includes support for bandwidth adaptation, NATs/firewalls, and heterogeneous node capabilities. Nodes in ESM continuously monitor their performance, and if it is not satisfactory, they will try to find a better parent in the tree. They do so by probing other nodes they know, and then choosing one of them as the new parent based on the probing results, using various criteria such as the delay, bandwidth, and saturation level of the probed candidates. Nodes employ a gossip-based mechanism to learn and propagate the existence of other peers in the group. To be more specific, each node A periodically (every 1.5 seconds) picks another node B from its routing table at random, and sends it a subset of group members that it knows. B adds to its routing table any members that it did not already know, and may send messages to the new nodes as part of normal protocol operations. Note that all ESM control messages use UDP.

Vulnerability: The gossip mechanism ESM employs to propagate membership information may be exploited, by having malicious nodes trick benign nodes into sending protocol packets to a victim. The victim can be an arbitrary Internet host (may not participate in ESM). A malicious node M in ESM could generate gossip messages that contain false information of the victim being part of the group. Benign nodes who receive the

fake gossip messages will include the victim in their routing tables, and later may send protocol packets to the victim as part of normal operations. Such packets become unsolicited traffic to the victim. The attack can be magnified if malicious nodes push fake gossip messages to benign nodes at a higher rate.

3 DDoS attacks exploiting P2P systems

We now discuss in detail how the vulnerabilities in Kad and ESM may be exploited to create large-scale DDoS attacks.

3.1 Attack using Kad

We present a set of attack heuristics that can lead to high attack amplifications with Kad, as follows:

- *Baseline:* As described in Fig. 1a, node A may seek to locate the node whose ID is the closest to a target ID. As part of the search operation, it may send a query to a (malicious) node M that is known to it. M respond with the IP and port of the victim V along with a fake ID which is closer to the target ID. A then sends queries to V, as part of the normal search operation. A may contact V for other protocol operations as well.
- *Attraction:* The magnitude of the attack above is dependent on the frequency with which other nodes may contact the malicious node. In general, this is small given that the system may involve millions of nodes. However, with Kad, a malicious node may proactively push information about itself to a large number of nodes in the system, forcing them to add the node to their routing tables. While it is not clear to us whether this particular vulnerability in Kad is intrinsic to its design, in general this is an illustration of how a node can exploit a P2P system to populate itself in participating nodes. Preventing such exploits in general is a hard problem [32].
- *Multifake:* Better amplification can be achieved if the malicious node includes the victim's information several times in the response to a query. The key insight behind this heuristic is the distinction between the *physical identifier* of a participating node such as its IP address, and its *logical identifier*, the node ID in the DHT space. Kad, and indeed many P2P systems, are designed to allow a participating node to communicate with multiple logical identifiers even though they share the same physical identifier (IP address). This has several advantages, for instance, enabling distinct users behind the same Network Address Translator (NAT) to participate in the system, even though they share the same physical IP address. The *Multifake* heuristic exploits this to achieve large amplifications by having the attacker node redirect benign node to multiple logical identifiers, that all share the IP address of the victim. Further, it seeks to achieve even greater magnification by having the attacker node include itself in the query responses a small number of times, so that the benign nodes will be repeatedly attracted to the attacker and consequently redirected to the victim.

Results: We implemented the above heuristics in an aMule client (a Linux clone of eMule), and let our modified client (i.e. the attacker) join the *live* Kad network. The victim node is a machine in our laboratory, which does not participate in the Kad network. Each experiment runs for several hours, and we report on magnitudes of attack seen. The experiments employ 5 attacker nodes unless otherwise mentioned.

Figure 2 shows the traffic at the victim with the three heuristics. The X-Axis is the time since the start of the experiment, in hours. The Y-Axis is the amount of traffic seen in Mbps. From bottom to top, the first three curves correspond to the attack magnitudes with the given heuristics alone. The last curve corresponds to the combination of all the heuristics. High magnitudes of over 10Mbps are seen at the victim when all heuristics are turned on. It is also interesting to see that the entire set of heuristics is required to generate the high attack magnitudes, and any subset is insufficient.

Figure 3 shows the distribution of distinct (IP, port) pairs of benign nodes that were redirected to the victim

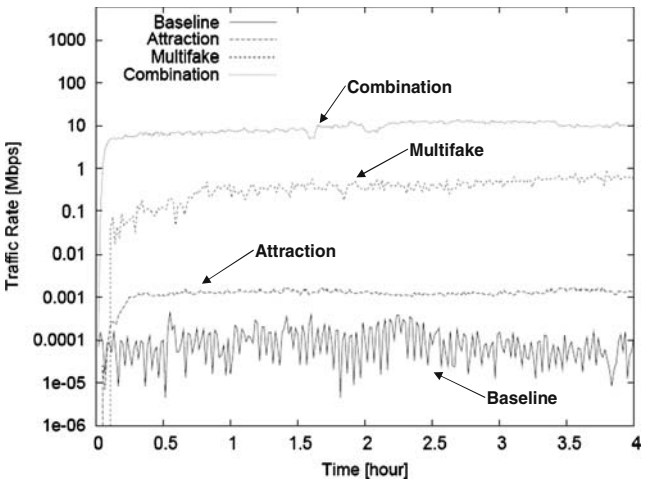


Fig. 2 Sensitivity to the heuristic employed by the attacker to increase the attack

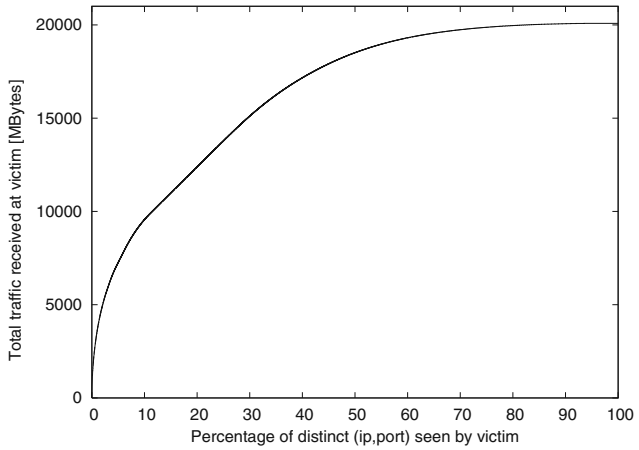


Fig. 3 CDF of the distinct (IP,port) pairs that have generated traffic at the victim

when all heuristics are turned on. A point (X,Y) in this graph means that traffic Y is contributed by X percent of distinct IP and port. Over 200,000 distinct (IP,port) pairs were redirected in the attack. As shown, the distribution is not sharply skewed indicating the traffic is not coming from any single node alone, which makes the attacks more difficult to contain.

Figure 4 shows the traffic observed at one attacker node, in terms of both traffic sent and received. The Y-Axis is the traffic rate in Kbps. The X-Axis is time since the start of the experiment, in hours. The main observation is that the traffic seen at the attacker node is only about 250 Kbps, which while higher than what a node would normally see, is 40 times lower than the traffic seen by the victim. Even if the total traffic at all attacker nodes is considered, there is still a amplification factor of 8. A point to note is the spike at the start of the experiment. This is due to the *Attraction* heuristic where the attacker node attempts to insert itself in the routing tables of many other nodes. We discuss the implications in the next paragraph.

We considered whether even higher attack magnitudes are possible by employing larger number of attacker nodes and making the *Attraction* heuristic more aggressive. Figure 5 shows an attack generated using 200 attacker nodes scattered around PlanetLab, with the victim in our laboratory. Traffic of over 700 Mbps was received by the victim after 14 h of experiment. This far exceeds what we feared, and indicates the criticality and seriousness of the problem. We abandoned further experiments on this line given the seriousness of the attacks. An ISP of one of the attacker nodes was concerned whether the node was running a random port-scan attack. This was because each attacker contacted around 100,000 Kad nodes as part of the *Attraction* heuristic, and not all nodes responded since some of them were no longer in the system. While this offers hope that such attacks could be detected, it may be feasible to evade detection by reducing the rate at which malicious nodes spread information about themselves to others. Significant attack magnitudes may still be achieved, though it may take longer for the attacks to ramp up to these values. We have conducted (carefully controlled) experiments to confirm this observation.

3.2 Attack using ESM

We exploit the vulnerabilities described in Section 2.2. A malicious node M may send to benign nodes gossip messages that contain false information about the victim being part of the group. The benign nodes will include the victim in their routing tables, and send

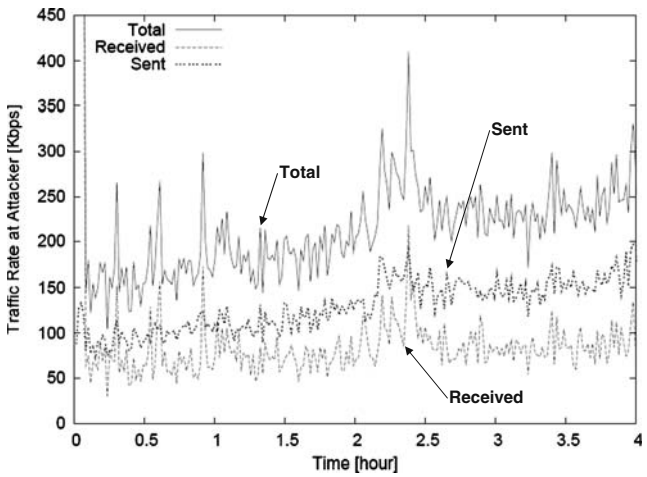


Fig. 4 Traffic seen at an attacker using all heuristics to generate the attack

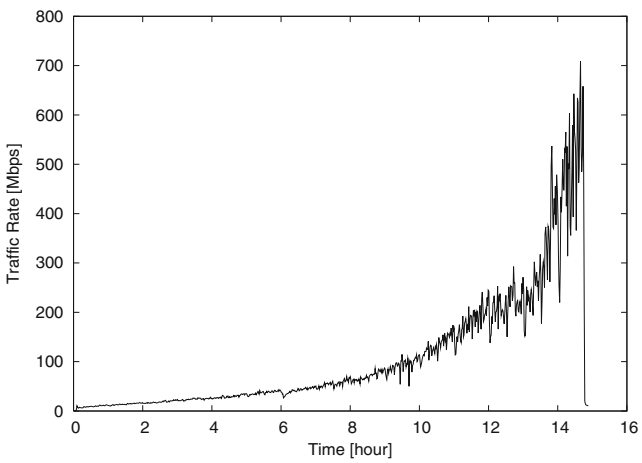


Fig. 5 Total traffic seen at victim, with 200 attackers, over a period of 15 h

protocol packets to the victim as part of normal operations. They may also propagate the victim to other benign nodes per the gossip protocol. Such an attack may be further magnified by the following heuristics:

- *Aggressive Push:* We exploit the *push-based* nature of the gossip protocol in ESM. In a push-based protocol, a member A may contact a member B in an unsolicited fashion, and disseminate membership information about another member C. In contrast, in a pull-based design (used in systems like Kad), B queries A, and accepts membership information only if it is in response to a prior query. A malicious node may exploit a push-based design by aggressively contacting many benign nodes in an unsolicited fashion, thereby spreading fake information at a high rate. Note that in pull-based designs, the ability of the malicious node to infect others is limited by the rate at which it is queried (though it could attract higher query rates if it could become popular).
- *Multifake:* Similar to *Multifake* in Kad, we augmented the heuristic to achieve greater attack amplifications by including the IP address of the victim several times in a gossip message, each time with a different logical identifier. ESM also makes use of logical identifiers (called uid in [10]) distinct from IP address and port information, primarily to handle issues with NATs. Like Kad, ESM allows a participating node to communicate with multiple logical identifiers even though they share the same physical identifier (IP address). Again this is motivated by NATs.

Results: We conducted controlled experiments on PlanetLab using the attack heuristics described above.

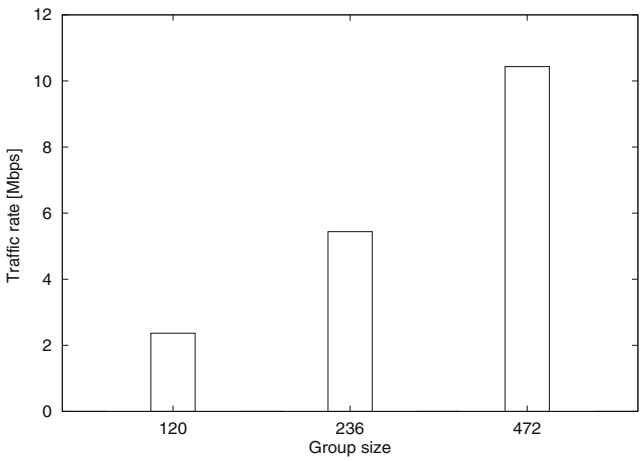


Fig. 6 Sensitivity to number of clients. Percentage of malicious clients fixed to 10%

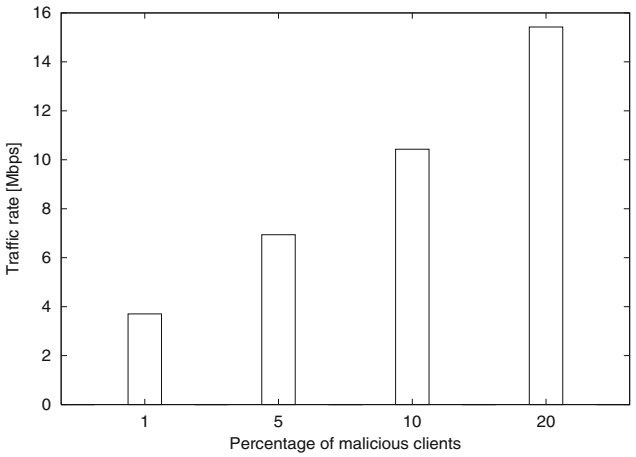


Fig. 7 Sensitivity to percentage of malicious clients. Total number of clients fixed to 472

Figure 6 shows the magnitudes of the DDoS attacks in comparison to the total number of ESM nodes. We fixed the percentage of attacker nodes to be 10% and varied the total number of nodes. The traffic seen by the victim is several Megabits per second, a factor of 1000 times more than the control traffic an ESM node would normally see (which is about 3 Kbps). Further, the attack traffic increases approximately linearly as the number of nodes increases. In real deployments which usually involve tens of thousands of participating nodes, the attack magnitude could be orders of magnitude higher. While the experiments above assume that 10% of the nodes are malicious, Fig. 7 plots the attack traffic fixing the total number of node at 472, and varying the percentage of attacker nodes. It shows that even a very small fraction of malicious nodes can cause a serious attack at the victim. For example, 1% of the nodes being malicious results in attack traffic of 4Mbps at the victim.

4 Discussion of possible mitigation schemes

In this section, we present possible mitigation schemes and discuss their strength and weakness.

4.1 Preferring pull-based membership propagation over push-based

A fundamental factor that impacts the vulnerability of membership management algorithms is whether they are push-based, or pull-based. In a push-based design, members may disseminate membership information to other members in an unsolicited fashion. In contrast, in a pull-based design, any information conveyed by a

member is always in response to a prior solicitation. Systems like ESM and CoolStreaming [10, 41] use push-based gossip algorithms, while BitTorrent and eMule use pull-based techniques.

We argue that pull-based protocols are preferable from the perspective of robust design since an attacker does not have control over the rate at which it can propagate malicious information. In contrast, push-based protocols are more vulnerable to compromise, since an attacker can control the rate at which it can redirect innocent participants to the victim. That said, a few points are in order. First, attacks where a node pushes malicious information at higher rates than normal in push-based approaches are potentially detectable since the amount of traffic that must be generated to spread false information is high. However, solutions for detection may not be straight-forward. They may either require all service providers of all nodes taking part in the P2P system to detect abnormal variations in traffic, or they may require correlating observations across multiple nodes in the system, neither of which are trivial. Second, pull-based algorithms may themselves not suffice. In particular, a factor dictating the vulnerability of pull-based algorithms is the ability of an attacker to attract queries from innocent participants towards itself. This may in turn depend on the number of nodes that know the attacker, as well as the skew in distribution of requests to any node—for example, arising due to variations in popularity of files owned by various nodes. In fact, our attacks on the Kad system leveraged a vulnerability which enabled an attacker to populate routing tables of a large number of innocent participants, thereby attracting queries towards itself. Finally, a subtle implementation issue with pull-based algorithms is that a member must be able to verify that any reply is actually in response to a prior request. This could be handled through a variety of well-known mechanisms. In fact, both BitTorrent and Kad handle this by storing transaction identifiers of outgoing pull requests and requiring that responses have identifiers matching outgoing requests.

4.2 Corroborating membership information through multiple sources

In order to limit the vulnerability of P2P systems to be exploited to launch DDoS attacks, when membership information is learned about a previously unknown node, it would be desirable to have a means of validating the information before using it.

One possible way to validate membership information is through active probing. In this way, when a node learns information about a peer node C, it directly probes C to ensure that C is a valid participant of the P2P system. No further protocol packet will be sent to C until a probe response from C is received. A potential concern with this approach is that this kind of validation can itself become subject to exploit if a large number of innocent participants try to conduct the validation. Further, the approach must be robust to probing failures caused by benign reasons such as churn, packet loss and NATs. We have explored this approach in a parallel work [37], and thus omit the discussion here.

In this paper, we explore an alternative means of validation that we term *Multi-Source Corroboration*, which does not rely on active probing. In this approach, a node will accept and communicate with a newly-learned peer node C only if it learns about C from at least k nodes, where k is a parameter. This approach has been used in several prior works on Byzantine-tolerant diffusion algorithms [21–23, 25]. These works assume that the maximum number of faulty nodes is known apriori, and require corroboration from at least 1 more than the number of faulty nodes. However the number of malicious nodes is unknown in our context. We thus adopt a more probabilistic variant, where the parameter k determines the number of sources from which corroboration is required. There is an interesting trade-off in setting the k value. From the security standpoint it is desirable to set k large, as the attacker needs to have at least k malicious nodes to mount an attack. However, the larger the k value, the longer it may take to receive responses from all k nodes, and can slow down convergence and performance. Clearly, the parameter k is critical—a higher value offers potentially stronger security properties, but slower convergence.

Theoretically an attacker can break such a scheme if he has control over k or more nodes. The attacker can make these malicious nodes lie about the same fake membership information, and defeat the corroboration. To mount such an attack the attacker must ensure that when a benign node does a search, it queries at least k malicious nodes.

Two common threat models are the attacker could launch a Sybil attack despite having only a small number of legitimate machines/IPs, or an attacker could control an entire prefix, say using BGP prefix hijacking. To protect against these obvious threats, we can modify the scheme so that a node may accept information about a previously unknown node C from any peer for the first time it learns about C, but for the remaining *k-1* times it only accepts information from peers that are each in a different prefix. In the rest of the paper, we use the terms “prefix” and “network” interchangeably.

Even with the above enhancement, an attacker can still defeat the scheme with machines in k or more prefixes. However it may not be entirely trivial to launch such an attack, because there is still need for the malicious nodes to become popular, to ensure that a large fraction of queries from innocent participants are each intercepted by at least k malicious nodes. We could further improve the resilience of the systems to such attacks, by accepting corroboration probabilistically, from a fraction f of randomly chosen addresses from the entire IP address space.

4.3 Bounding multiple references to the same network entity

Intuitively a node should stop sending further packets to a newly-learned peer, if it has not received any response for the first few packets sent to that peer. However, simply doing so is not sufficient to mitigate the DDoS attacks, because an attacker may keep redirecting an innocent participant to the same victim IP, each time with a different logical identifier, for example, using the *Multifake* heuristic presented in Section 3.1. Further, an easy way to generalize such attacks is to flood the access link of a *network*, by redirecting innocent participants to different IPs that all belong to the same victim network.

One way to solve the above problem, that we term *Bounding-References*, is to bound the number of references to a network entity that a node is willing to accept. We consider this scheme because a more recent release of the eMule/Kad software implemented such a bounding scheme, as a response to the attacks presented in our workshop paper [39]. In particular, in the new implementation a node would accept at most 1 logical identifier for a single IP, and at most 10 IPs belonging to the same /24 prefix.

The advantage of such a scheme is that it bounds the number of packets an innocent participant may send to a single victim IP or prefix. For the scheme to be effective, the bounds must be small. However a disadvantage is that there may be genuinely many participants sharing the same IP (for example, due to the use of NAT), or belonging to the same prefix (for example, a residential ISP with a lot of Kad users), thus the scheme may falsely prevent communication between genuine participants, and degrade their performance.

4.4 Other approaches

We next discuss two other possible approaches and potential issues with them. First, centralized authorities can simply certify membership information by providing signed certificates that indicate that a member belongs to a group. The certificates may be distributed through the membership management mechanisms, and enable a participant to verify the membership information corresponds to a genuine member. However, central authorities cannot be assumed in many deployments such as Kad and thus we only consider mechanisms that do not rely on their existence.

Second, it may be feasible to leverage the properties of DHTs, and design solutions tailored to them. For instance, one approach is to assign each node an ID dependent on its IP address, for instance the hash of its IP address [34]. An attacker that attempts to provide fake membership information in response to a search query must then ensure that (i) the victim ID obeys the necessary relationship to the victim IP; and (ii) the victim ID is close to the target ID of the search. These dual constraints on the victim ID may be difficult to simultaneously satisfy, complicating the attack. We note that a complete solution on these lines must address several issues such as the need to accommodate multiple participants behind the same NAT which share a common IP address, and the fact the victim ID is only loosely constrained by the target ID. While the approach has promise, we do not explore it further since our focus in this paper is on mechanisms that apply to both unstructured approaches and DHT-based approaches.

5 Evaluation methodology

We next present evaluations exploring the strength and weakness of the schemes we presented in Section 4, and quantifying security and performance trade-offs involved with design heuristics. In this section, we describe our evaluation goals, metrics and experimental methodology. We present the results in next section.

5.1 Evaluation goals

Our evaluations are motivated by several goals:

- How effective are pull-based mechanisms in lowering attack magnitudes as compared to push-based mechanisms?
- How does parameter *"k"* in the *Multi-Source Corroboration* scheme impact application performance? We recall that k is the number of distinct sources that must verify membership information before it is used.

- How small the bounds in the *Bounding-References* scheme can be? How prevalent is the use of multiple logical identifiers for the same IP in actual deployments? How many participants are from the same network prefix?

5.2 Performance metrics

We characterize the performance and security trade-offs in our heuristics as follows:

- *Application performance:* For file distribution (Kad), we consider the time taken for a search for a given target ID to be successful. Since some searches may not be successful at all, the fraction of searches that are successful is also considered. For video broadcast (ESM), we consider the fraction of the streaming video rate received by participating nodes.
- *Attack Impact:* We evaluate the impact of an attack on a victim by considering the rate of traffic received at the victim.

5.3 Methodology

To evaluate the mitigation schemes, we have implemented them in the ESM and Kad systems, and conducted evaluations on PlanetLab. We also conducted measurement studies of the live Kad network.

ESM experiments: Our experiments with ESM leveraged traces from real broadcast events [10] to model the group dynamics patterns, and bandwidth-resource constraints of nodes. We emulate twenty minute segments of the trace. The clients already present in the trace at the start of the segment join in a burst over the first two minutes, then follow join/leave patterns exactly as in the trace for the next 20 min. For the trace segment used, 363 nodes participate, with a group peak size of 108 nodes. The streaming video rate employed was 475 Kbps, which represents typical media streaming rates in real settings like [10]. We used two versions of the attacker. In the first version, the attacker behaved similar to a regular node, except that whenever it needed to push membership information (or was contacted by another member pulling membership information), fake information regarding the victim was sent. We term such an attacker *undetectable-attacker*, since the traffic patterns it introduces are not distinguishable from a normal node in the system. We also considered a second attacker that we term *aggressive-attacker,* which periodically pushes fake membership information about the victim to a large number of nodes in the system. This kind of attacker only impacts a push-based solution. We note that an aggressive-attacker may be sending more traffic than regular nodes in the system.

Kad experiments: Our experiments with Kad used a synthetic trace of join/leave patterns where the inter-arrival patterns of nodes, and the stay time duration of nodes followed a Weibull distribution. This was based on recommendations by a recent measurement study [36]. The trace has 1455 clients in total, with peak group size around 300. Nodes executed a search pattern where each node periodically (every 30 s) conducted a search for a random logical identifier, which was intended to simulate a search request for a file. We assumed that the search successfully reached an index-node if it reached any of the 5 nodes whose IDs are closest to the logical identifier for which the search was conducted. The attackers employed all the heuristics described in Section 3.1.

Kad measurement study: To understand the prevalence of multiple logical identifiers sharing the same IP, and multiple IPs belonging to the same network in actual deployments, we have conducted a measurement study of the real Kad infrastructure. We implemented a Kad crawler which basically issues random searches on the Kad network and records the results returned by the network. The crawler collected 3.5 million distinct <Logical identifier, IP, Port> triples distributed across 487284 /24 prefixes in a 20 hour time period. Further, every second the crawler probes 10 clients it recently learned, and records the responses.

6 Results

Section 6.1 discusses potential benefits of pull-based approaches over push-based ones. Since ESM uses push, and Kad uses pull, our experiments in this section employ ESM. Section 6.2 focuses on security and performance trade-offs in the *Multi-Source Corroboration* scheme. Section 6.3 shows issues with designing the bounds for the *Bounding-References* scheme. The last two sections are relevant to both Kad and ESM, and we focus our evaluations on Kad.

6.1 Pull vs. push approaches

We present evaluation results of the base ESM system with push-based mechanisms, as well as our refinements that employ pull-based mechanisms. Our evaluations are conducted using both the undetectable-attacker and the aggressive-attacker described in Section 5.3. Figure 8 plots the average traffic seen at the victim and

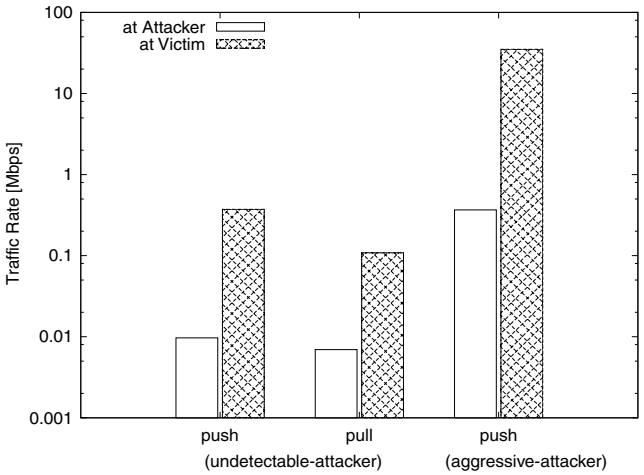


Fig. 8 Traffic seen by victim and attacker, for undetectable and aggressive attackers

an attacker during the run, for both the undetectable-attacker, and the aggressive-attacker. Further, for the undetectable-attacker, traffic is shown for both pull and push cases and for the aggressive-attacker, traffic is shown only for the push case since pull is not relevant here. We note that the traffic at the attacker is extremely small for the undetectable-attacker but about 300 Kbps for the aggressive-attacker. This is consistent with the amount of work the attacker does in each case. We also note that the traffic at the attacker, for all cases, is at least one order of magnitude less than the traffic at the victim, which demonstrate the potency of the attack.

From the perspective of the victim, there are two points to note from the graph. First, the magnitude of attack traffic can be very high with push-based approaches when an aggressive attacker is employed, which may be prevented by pull-based approaches. Second, even with an undetectable-attacker, the magnitude of attack traffic with pull-based mechanisms is slightly less than push-based mechanisms. This is because with push-based techniques, an attacker may always infect an innocent participant in any iteration—in pull-based techniques, this is dependent on the probability with which an innocent participant contacts an attacker node.

Discussion: We believe where possible, it is desirable to use pull-based mechanisms to limit the rate at which attackers may infect innocent participants. However, simply preferring pull-based approaches is not sufficient. First, use of push may be difficult to avoid in some cases. For example, when a node is a source of a file, it needs to publish this information to other nodes, inherently a push-based operation. Second, even in a pull-based protocol, there may still be mechanisms by which an attacker can attract more queries from innocent participants towards itself, perhaps because of the skew in distribution of requests to any node—for example, arising due to variations in popularity of files owned by various nodes. In fact, the search mechanism in Kad is a pull-based one, but we were still able to exploit it to cause DDoS attacks of high magnitude, by attracting a large number of queries towards the attacker nodes (Section 3.1).

6.2 Corroborating membership information through multiple sources

We now consider the trade-offs involved in the *Multi-Source Corroboration* scheme. The key parameter for the scheme is k, the number of sources from which corroboration is required. It is clear that the greater the k value, the higher the bar for the attacker.

We first evaluate the effectiveness of the scheme in mitigating DDoS attacks, using the methodology described in Section 5.3. Figure 9 shows the traffic generated at the victim as a function of the number of attackers. There are two sets of bars, each corresponding to a setting with a particular number of attackers. Each set has 3 bars corresponding to different k values. A k value of 1 indicates the default Kad system. Even in the presence of a moderate number of attackers, the traffic at the victim is high for $k = 1$—about 6 Mbps for 6 attackers. Using $k = 2$ is effective in reducing the attack traffic. For example, the traffic at the victim is about 89 Kbps for settings with 10 attackers. Using a higher value of k further reduces the traffic.

While this shows potential benefits of *Multi-Source Corroboration* in reducing attack magnitudes, we note that the actual magnitude depends on the specific attack

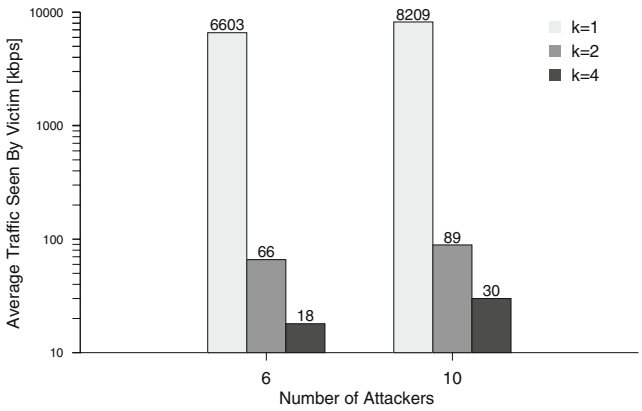


Fig. 9 Traffic seen by victim as a function of the number of attackers. Traffic is averaged across the whole run

heuristics used. It may be possible to get larger magnitudes with more sophisticated attack heuristics than those described in Section 3.1. To guarantee resistance to attacks, it is best to have a k value greater than the number of attacker nodes.

While larger k values are more desirable from the security stand point, one important concern is application performance. The larger the k value, the longer it may take to receive responses from all k nodes, and can slow down convergence and performance. We next focus our evaluation on the impact of k on application performance, in the absence of attackers.

Table 1 summarize our evaluation results. The first row shows the success ratio, or the fraction of searches that are successful. While a value of $k = 1$ (i.e. the original Kad client) has a success ratio of close to 1, $k = 2$ performs well too, with a success ratio of 0.98. However, the success ratio with $k = 4$ is much lower, only around 0.7. Investigation reveals that the reason the success ratio drops significantly with $k = 4$ is that, either nodes did not hit enough number of peers to confirm membership information from during the search process, or the "wait-time" involved in confirming the membership information from so many sources was so long that the search timed out before enough confirmations were received.

While the success ratio is one metric of performance, the second row of Table 1 shows the delays incurred by successful searches. For any setting, all searches conducted across all nodes are considered, and the average delays are computed. The results are consistent with the success ratio metric. $k = 1$ performs the best with average delays across searches less than 1 s. The performance of $k = 2$ is slightly worse, but still acceptable, with average delays around 2 seconds. With a value of $k = 4$ however, the average delays are high, and over 13 s.

Discussion: These results indicate that, the performance penalty incurred by *Multi-Source Corroboration* is modest only when the k value is small. Even with a slightly higher k value such as $k = 4$, the performance degradation becomes unacceptable. This in turn suggests that the *Multi-Source Corroboration* scheme works best in attack scenarios where the attacker controls a single machine, or machines reside in a single prefix. In such scenarios, a k value of 2 is sufficient to prevent the attacks, while achieving satisfactory application performance. However, such a scheme is not suitable in regimes where the attacker machines span many prefixes, as for such regimes it is necessary to set the k value to be greater than 2, but our results show that for such k values the cost on performance would be more substantial.

Table 1 Comparing the impact of different k values on application performance

Metrics	$k = 1$	$k = 2$	$k = 4$
Success ratio	99%	98%	73%
Delay	0.6 s	2.2 s	13.2 s

6.3 Bounding multiple references to the same network entity

Recall that this approach bounds the number of references to a network entity that a node is willing to accept. More specifically, it bounds the number of IDs associated with the same IP address, and the number of IPs sharing the same prefix. By doing so it bounds the number of packets an innocent participant may send to a single victim IP or prefix. An important question with such an approach is how the bounds should be set. We try to answer this question by analyzing a trace collected by crawling the live Kad network using the methodology described in Section 5.3.

Figure 10 presents results from the analysis of the traces. There are 3 sets of bars, each corresponding to a different notion of a physical identifier—(IP,port) pair, IP, and /24 prefix. For each set, there are 3 bars, corresponding to 1, 5, and 10 logical identifiers. Each bar shows the fraction of distinct physical identifiers that are associated with a certain number of logical identifiers, or less. We make several observations.

First, close to 100% of the (IP,port) pairs involve only 1 logical identifier. Second, when IP alone is considered as the physical identifier, over 92% of the IPs are associated with only 1 logical identifier, and over 99.8% of the IPs are associated with 10 logical identi-

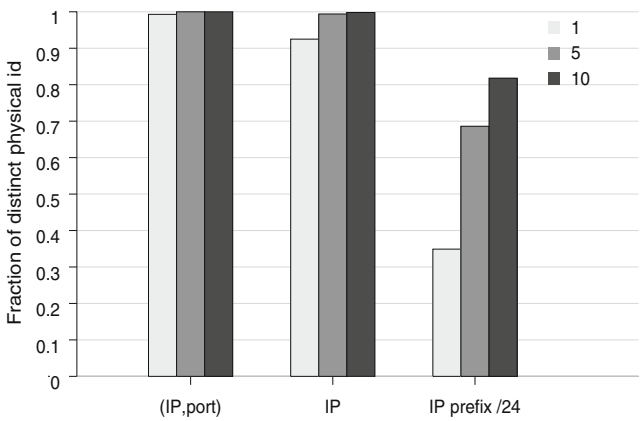


Fig. 10 Fraction of distinct physical identifiers associated with a certain number of logical identifiers, or less

fiers or less. These results indicate that, while setting the bound of logical identifiers sharing the same IP to 1 may be a little too strict, it is reasonable to set it to a small value such as 3. Third, Fig. 10 also shows the fraction of 24-bit IP prefixes that are associated with a certain number of logical identifiers (or less). Only 35% of the prefixes are associated with 1 client, and about 20% of the prefixes are associated with more than 10 distinct clients. This indicates that larger bounds may be required for prefixes.

The bounds for prefixes could probably be smaller if there is significant stale information and not all nodes returned by the crawler are alive at a given time. To gain more insight into the liveness of nodes, Fig. 11 plots the success rate of probes our crawler sent to various /24 prefixes. We only consider prefixes to which 10 or more probes were sent. 90% of the prefixes responded at least half the probes, and 50% of the prefixes responded to more than 90% probes. The result shows that most nodes in our trace were indeed alive at the time of crawling, suggesting that the bounds for prefixes cannot be substantially lowered.

Our analysis so far considers fixed prefix length (/24), motivated by a recent release of the eMule/Kad client software (see Section 4.3). However, this makes the scheme vulnerable to attacks on networks with coarser granularities (i.e. shorter prefix lengths). To evaluate whether a simplifying approximation of /24 is reasonable, we examined the RouteView database which contains prefix and netmask information extracted from BGP routing table snapshots [2, 19] in order to determine the appropriate prefix lengths, and summarize our results in Fig. 12. It shows clearly that an approximation

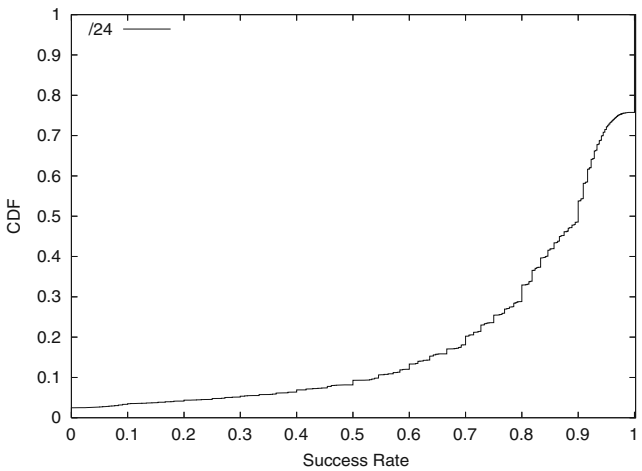


Fig. 11 CDF on the success rate of probes to various /24 prefixes in the real Kad system

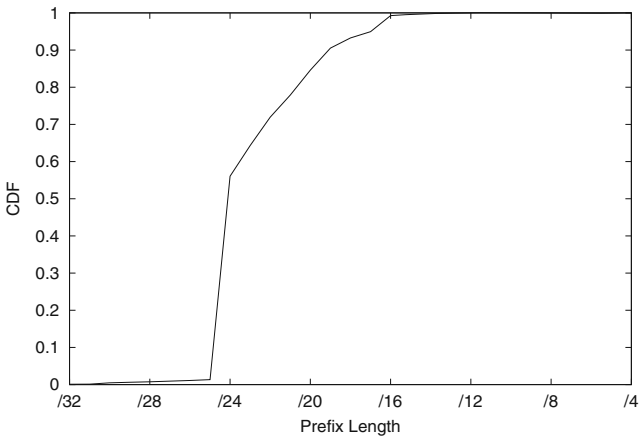


Fig. 12 CDF on prefix lengths, obtained from a database of BGP routing table snapshots

of /24 is far from accurate, as 40% of the networks have smaller prefix lengths. Given this result, we believe that it is important to group IPs based on such a database, rather than grouping them statically using /24.

We analyze our trace again based on the above results. This time we focus on the prefix level, and use the RouteView database to determine prefix lengths. The result is shown in Fig. 13. It shows that 33% of the prefixes have 10 or more IPs participating in Kad, 15% of the prefixes have 50 or more IPs participating, and 9% have 100 or more. This again indicates that the bounds for prefixes must be set sufficiently high so that the performance will not likely to be affected. However, setting the bounds high makes the scheme less effective in mitigating DDoS attacks.

Discussion: While the *Bounding-References* scheme can limit DDoS attacks, the concern is that it may

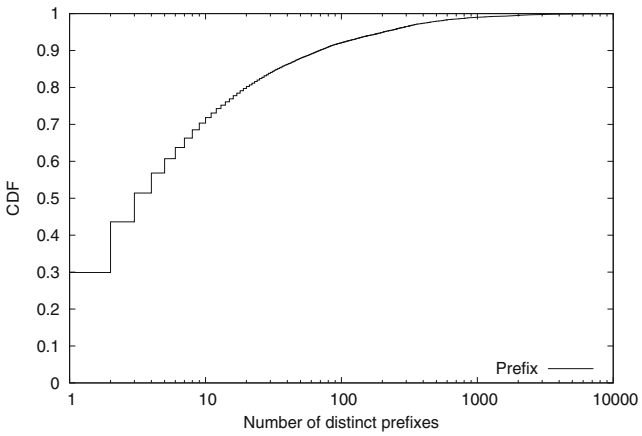


Fig. 13 Fraction of distinct prefixes associated with a certain number of IPs, or less

also limit the communication with genuine participants sharing the same IP or network prefix. Our results reveal that, while limiting communication with participants sharing the same IP is likely to have negligible impact, the impact of limiting participants belonging to the same prefix can be more significant.

7 Related work

It was first observed in [26] that the intrinsic characteristics of P2P systems could be exploited for indirection attacks. Since then, a few other recent works including our own workshop paper [6, 11, 14, 30, 39] have shown the feasibility of exploiting P2P systems to launch DDoS attacks. Ours is the first work to show the feasibility of exploiting Kad, as well as first to show that such attacks are also possible with P2P systems used for video broadcasting. The attack magnitudes and amplification reported in our paper is much higher indicating the seriousness of the threats. Further, we have discussed and evaluated several possible mitigation schemes. Ours is the first work to systematically study defenses to the best of our knowledge.

Naoumov and Ross [26] presents an attack on Overnet, another DHT-based system. The attack relies on a attacker impersonating the victim when talking to an innocent client, which forces the client to communicate with the victim. Further, due to an implementation vulnerability in Overnet, spoofing at the IP source address level is not required, but merely at the application payload level. The paper also proposed another type of DDoS attacks in which the attacker tries to falsely publish the victim as sources for popular files so other peers will try to download the files by initiating TCP connections to the victim. In contrast, the attacks we present exploit issues with push-based protocols, having attackers attract query traffic, and achieving amplification through multiple logical identifiers being mapped to the same physical identifier. The attack magnitudes we report are also orders of magnitude higher than those reported in [26].

Defrawy et al. [11] showed the feasibility of DDoS attacks by exploiting BitTorrent [7]. In BitTorrent, a central entity called tracker, will keep track of peers downloading a file and will distribute these peers to others that have the complete file or pieces of it. The tracker contact information is in the torrent file which can be downloaded from popular websites. In [11], the authors propose to include the victim's information in the torrent file as one of the trackers and let the clients aggressively contact the victim as part of the protocol. Then, if the file to be downloaded is popular enough, many innocent clients will contact the victim thinking it is a tracker. This can generate a large number of TCP connections at the victim and produce a DDoS attack. In contrast, we focus on decentralized P2P applications, a DHT-based file-sharing system (Kad) and a gossip-based video broadcast system (ESM), where malicious clients drive innocent clients to attack the victim.

In another work [37], we have dissected all known DDoS attacks that exploit P2P systems [6, 11, 14, 26, 30, 39], and categorized them based on the underlying cause for attack amplification. We have explored a solution where nodes validate membership information through active probing, and address challenges associated with the approach. These challenges include ensuring that the probes used for validation do not themselves become a source of DDoS, and ensuring that the scheme is robust to benign validation failures due to churn, packet loss and use of NATs. In contrast, in this paper we focus on details of attacks on two specific systems, and explore solutions that do not involve active probing.

Other works have explored attacks caused by DNS and web-server reflectors, and misuse of web-browsers and bot-nets [1, 18, 20, 27]. We believe the rich and complex design space of P2P systems makes them worthy of study in their own right. Several works [8, 12, 32, 33, 38] focus on how malicious nodes in a P2P system may disrupt the normal functioning, and performance of the overlay itself. In contrast, we focus on DDoS attacks on the external Internet environment. Our work benefits from work on the design of byzantine resilient gossip protocols in the traditional distributed systems community [21–23, 25]. While we leverage insights from these works, we believe the scalability, heterogeneity and performance requirements with P2P networks and applications pose unique challenges and it is necessary to investigate the issues in the context of actual systems. Finally [31] has looked at detecting faults, anomalies and security vulnerabilities in overlays using query processing.

8 Conclusions

In this paper, we have made two contributions:

- First, we have shown that it is feasible to exploit P2P systems to launch DDoS attacks with high magnitudes and large amplification. We created attacks of several hundred megabits per second and

with an amplification factor of 40, by exploiting a mature and widely deployed P2P file-sharing system and an extensively deployed video broadcast system. The systems represent contrasting applications, and involve different and representative membership management designs.

- Second, we evaluate the potential of three mitigation schemes in enhancing the resilience of Kad and ESM. The evaluation is done through both experiments on the PlanetLab and measurement to the real systems. Our findings include:

  (i) It is desirable to use pull-based protocols where possible to limit the rate at which attackers may infect innocent participants. However, simply preferring pull-based approaches is not sufficient because in some cases the use of push may be difficult to avoid, and even in a pull-based protocol, there may still be mechanisms by which an attacker can attract more queries from innocent participants towards itself.

  (ii) The *Multi-Source Corroboration* scheme works best in attack scenarios where the attacker machines span a small number of prefixes. As the attacker machines span more prefixes, it is necessary to increase the value of parameter k, the number of distinct sources that corroboration must be obtained from. However, our results show that even for a moderate k value such as 4 the cost on performance would be substantial.

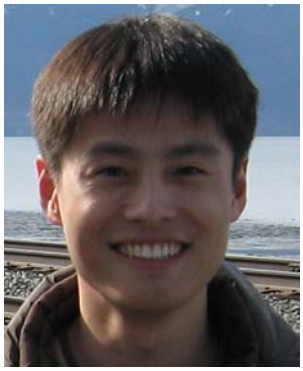
  (iii) The *Bounding-References* scheme limits DDoS attacks. For the scheme to be effective, the bounds must be small. However, small bounds may limit a node's ability to communicate with genuine participants that share the same IP or network prefix. Our results reveal that, while limiting communication with participants sharing the same IP is likely to have negligible impact, limiting communication with participants belonging to the same prefix may have more significant impact.

Overall, our results shed light on the potential of the schemes, in what scenarios and regimes they may be useful, and the trade-offs associated with them. We believe this work has taken a first but important step towards understanding the interplay between membership management mechanisms, and the feasibility of exploiting P2P systems to cause DDoS attacks.

References

1. Vaughn R, Evron G (2008) DNS amplification attacks. http://www.isotf.org/news/DNS-Amplification-Attacks.pdf
2. University of Oregon (2008) Route views project. http://www.routeviews.org
3. Microsoft (2008) Windows peer-to-peer networking. http://www.microsoft.com/p2p
4. Ali S, Mathur A, Zhang H (2006) Measurement of commercial peer-to-peer live video streaming. In: Proc. WRAIPS, Waterloo, August 20006
5. aMule (2008) aMule homepage. http://www.amule.org/
6. Athanasopoulos E, Anagnostakis KG, Markatos E (2006) Misusing unstructured p2p systems to perform dos attacks: the network that never forgets. In: Proc. ACNS, Singapore, 6–9 June 2006
7. BitTorrent (2008) BitTorrent homepage. http://www.bittorrent.org
8. Castro M, Druschel P, Ganesh A, Rowstron A, Wallach DS (2002) Security for structured peer-to-peer overlay networks. In: Proc. OSDI, Boston, 9–11 December 2002
9. Cho K, Fukuda K, Esaki H (2006) "The impact and implications of the growth in residential user-to-user traffic". In: Proc. ACM SIGCOMM, Pisa, 11–15 September 2006
10. Chu Y, Ganjam A, Ng TSE, Rao SG, Sripanidkulchai K, Zhan J, Zhang H (2004) Early experience with an internet broadcast system based on overlay multicast. In: Proc. USENIX, Boston, 27 June–2 July 2004
11. Defrawy KE, Gjoka M, Markopoulou A (2007) BotTorrent: misusing BitTorrent to launch DDoS attacks. In: Proc. Usenix SRUTI
12. Douceur J (2002) The Sybil attack. In: Proc. IPTPS
13. EMule (2008) EMule homepage. http://www.emule-project.net
14. Harrington J, Kuwanoe C, Zou C (2007) A BitTorrent-Driven distributed denial-of-service attack. In: Proc. of 3rd international conference of security and privacy in communication networks (SecureComm)
15. Hei X, Liang C, Liang J, Liu Y, Ross K (2006) Insights into pplive: a measurement study of a large-scale p2p iptv system. In: Proc. workshop on internet protocol TV (IPTV) services over world wide web in conjunction with WWW2006
16. Huang Y, Fu TZ, Chiu D-M, Lui JC, Huang C (2008) Challenges, design and analysis of a large-scale p2p-vod system. In: SIGCOMM '08: proceedings of the ACM SIGCOMM 2008 conference on data communication. ACM, New York, pp 375–388
17. Mirkovic DDJ, Dietrich S, Reiher P (2005) Internet denial of service: attack and defense mechanisms. Prentice Hall, Englewood Cliffs
18. Kandula S, Katabi D, Jacob M, Berger A (2005) Botz-4-sale: surviving organized DDoS attacks that mimic flash crowds. In: Proc. NSDI

19. Krishnamurthy B, Wang J (2000) On network-aware clustering of web clients. SIGCOMM Comput Commun Rev 30(4):97–110
20. Lam V, Antonatos S, Akritidis P, Anagnostakis KG (2006) Puppetnets: misusing web browsers as a distributed attack infrastructure. In: Proc. of ACM CCS
21. Malkhi D, Mansour Y, Reiter M (2003) Diffusing without false rumors: on propagating updates in a byzantine environment. Theor Comp Sci 299(1–3):289–306
22. Malkhi D, Pavlov E, Sella Y (2001) Optimal unconditional information diffusion. In: 15th international symposium on distributed computing (DISC 2001)
23. Malkhi D, Reiter M, Rodeh O, Sella Y (2001) Efficient update diffusion in byzantine environments. In: 20th symposium on reliable distributed systems (SRDS 2001) .
24. Maymounkov P, Mazieres D (2002) Kademlia: a peer-to-peer information system based on the xor metric. In: Proc. IPTPS
25. Minsky Y, Schneider F (2003) Tolerating malicious gossip. Distrib Comput 16(1):49–68
26. Naoumov N, Ross K (2006) Exploiting p2p systems for DDoS attacks. In: International workshop on peer-to-peer information Management
27. Paxson V (2001) An analysis of using reflectors for distributed denial-of-service attacks. Comput Commun Rev 31(3)
28. Ratnasamy S, Francis P, Handley M, Karp R, Shenker S (2001) A scalable content-addressable network. In: Proceedings of ACM SIGCOMM
29. Rowstron A, Druschel P (2001) Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM international conference on distributed systems platforms (Middleware)
30. Sia KC (2007) DDoS vulnerability analysis of BitTorrent protocol. Technical report, UCLA
31. Singh A, Maniatis P, Roscoe T, Druschel P (2006) Using queries for distributed monitoring and forensics. In: Proc. of EuroSys
32. Singh A, Ngan T-W, Wallach D (2006) Eclipse attacks on overlays: threats and defenses. In: Proc. INFOCOM
33. Sit E, Morris R (2002) Security considerations for peer-to-peer distributed hash tables. In: Proc. IPTPS
34. Stoica I, Morris R, Karger D, Kaashoek, MF, Balakrishnan H (2001) Chord: a scalable peer-to-peer lookup service for internet applications. In: Proceedings of ACM SIGCOMM
35. Stutzbach D, Rejaie R (2006) Improving lookup performance over a widely-deployed DHT. In: Proceedings of IEEE INFOCOM
36. Stutzbach D, Rejaie R (2006) Understanding churn in peer-to-peer networks. In: Proc. ACM SIGCOMM IMC
37. Sun X, Torres R, Rao S (2007) Preventing DDoS attacks with P2P systems through robust membership management. Technical report
38. Wallach D (2002) A survey of peer-to-peer security issues. In: International symposium on software security
39. Sun X, Torres R, Rao S (2007) DDoS attacks by subverting membership management in P2P systems. In: Proc. NPSec
40. xMule (2008) xMule homepage. http://www.xmule.ws/
41. Zhang X, Liu J, Li B, Yum T-SP (2005) DONet/CoolStreaming: a data-driven overlay network for live media streaming. In: Proceedings of IEEE INFOCOM
42. Zhao B, Huang L, Stribling J, Rhea SC, Joseph AD, Kubiatowicz J (2004) Tapestry: a resilient global-scale overlay for service deployment. IEEE J Sel Areas Commun 22(1):41–53



Xin Sun is a PhD student in the School of Electrial and Computer Engineering at Purdue University. He works with Prof. Sanjay Rao. He received his B. Eng. degree from University of Science and Technology of China in 2005. His research interests are in overlay networks with an emphasis on security.

Ruben Torres is a Ph.D. student in the School of Electrical and Computer Engineering at Purdue University and a research assistant in the Internet Systems Lab. Ruben received his MS in electrical and computer engineering from Purdue University in 2006 and his BS in electronics and communications engineering from the University of Panama in 2002. His research interests include distributed systems and security, with an emphasis on peer-to peer networks.

Sanjay G. Rao received the Bachelor's degree in Computer Science and Engineering from the Indian Institute of Technology, Madras in 1997 and the Ph.D from the School of Computer Science, Carnegie Mellon University in 2004. He is an Assistant Professor in the School of Electrical and Computer Engineering, Purdue University, West Lafayette, where he leads the Internet Systems Laboratory. He was a visiting researcher in the Network Measurement and Management group at AT&T Research, Florham Park, New Jersey in Summer 2006. He wrote the first paper to show the viability of an overlay approach to multicast, and led the design of an overlay broadcasting system based on a peer-to-peer architecture. His research interests are in Networking, and Distributed Systems. His current projects are in Peer-to-Peer systems, and Network Management. Prof. Rao has served on the Technical Program Committees of several workshops and conferences including ACM SIGCOMM, IEEE Infocom, and ACM CoNEXT.