Denial-of-Service Resilience in Peer-to-Peer File Sharing Systems*

D. Dumitriu,† E. Knightly,‡ A. Kuzmanovic,* I. Stoica,** and W. Zwaenepoel†

†EPFL, Lausanne, Switzerland
‡Rice University, Houston, Texas
*Northwestern University, Evanston, Illinois
**University of California at Berkeley

ABSTRACT

Peer-to-peer (p2p) file sharing systems are characterized by highly replicated content distributed among nodes with enormous aggregate resources for storage and communication. These properties alone are not sufficient, however, to render p2p networks immune to denial-of-service (DoS) attack. In this paper, we study, by means of analytical modeling and simulation, the resilience of p2p file sharing systems against DoS attacks, in which malicious nodes respond to queries with erroneous responses. We consider the file-targeted attacks in current use in the Internet, and we introduce a new class of p2p-network-targeted attacks.

In file-targeted attacks, the attacker puts a large number of corrupted versions of a *single* file on the network. We demonstrate that the effectiveness of these attacks is highly dependent on the clients' behavior. For the attacks to succeed over the long term, clients must be unwilling to share files, slow in removing corrupted files from their machines, and quick to give up downloading when the system is under attack.

In network-targeted attacks, attackers respond to queries for *any* file with erroneous information. Our results indicate that these attacks are highly scalable: increasing the number of malicious nodes yields a hyperexponential decrease in system goodput, and a moderate number of attackers suffices to cause a near-collapse of the entire system. The key factors inducing this vulnerability are (i) hierarchical topologies with misbehaving "supernodes," (ii) high path-length networks in which attackers have increased opportunity to falsify control information, and (iii) power-law networks in which attackers insert themselves into high-degree points in the graph.

Finally, we consider the effects of client counter-strategies such as randomized reply selection, redundant and parallel download, and reputation systems. Some counter-strategies (e.g., randomized reply selection) provide considerable immunity to attack (reducing the scaling from hyperexponential to linear), yet significantly hurt performance in the absence of an attack. Other counter-strategies yield little benefit (or penalty). In particular, reputation systems show little impact unless they operate with near perfection.

Categories and Subject Descriptors

C.2.0 [**Security and Protection**]: Denial of Service;
C.2.2 **[Computer-Communication Networks]:** Network Protocols

General Terms

Algorithms, Performance, Theory, Security

Keywords

Peer-to-peer, denial of service, file pollution, network-targeted attacks

1. INTRODUCTION

Peer-to-peer (p2p) file sharing networks can be subjected to intense Denial-of-Service (DoS) attacks. For example, it has been reported that the music industry places false content on p2p networks used for trading copyrighted music [3, 7, 16]. Likewise, recording artists have released false content on p2p networks [7, 16, 21]. On one hand, one may expect that p2p file-sharing systems are robust to DoS attacks, because popular data is highly replicated and system resources such as bandwidth and storage are immense and widely distributed. On the other hand, one may expect a p2p network whose topology is characterized by a power-law graph to be vulnerable to attack [2].

The contributions of this paper are to identify the key factors that affect the DoS resilience of a p2p file sharing system and to quantify the impact of these factors via analytical modeling and simulation. These factors include protocol properties (*e.g.*, hierarchy via "supernodes"), graph properties (*e.g.*, power-law vs. k-regular graphs), client counter-DoS strategies (*e.g.*, parallel download and randomization strategies), and user-behavior factors (*e.g.*, willingness to share files and persistence in downloading a file when the system is under a DoS attack). Thus, our findings provide critical guidelines for DoS-resilient design of p2p architectures, protocols, and client counter-strategies by characterizing attack scalability and even "collapse points" associated with each design decision.

*E. Knightly is supported by NSF ITR grant ANI-0331620 and by a gift from Hewlett Packard. I. Stoica is supported by NSF grant ANI-0225660.

SIGMETRICS'05, June 6–10, 2005, Banff, Alberta, Canada.

Scope of Attacks. We consider known file-targeted attacks targeted against popular files [7, 16], and we introduce a new class of more devastating attacks against entire p2p file sharing systems. In file-targeted DoS attacks, a malicious node advertises a corrupted (polluted) copy of a given file, and distributes this copy when chosen by another peer. Both measurements [7, 16] and anecdotal evidence [3, 21] indicate that the music industry is depositing large volumes of polluted files into p2p file sharing systems such as KaZaA. Moreover, companies such as Overpeer[1] or Retsnap[2] publicly offer their pollution-based DoS services to the entertainment industry for protecting copyrighted materials.

Next, we develop and study a new class of attacks designed to collapse a p2p network's goodput. In such an attack, a malicious peer modifies replies to queries for *any* file, before it forwards them to the client. In a "false reply attack", the malicious peer points the client to itself. When the client then requests a download from the malicious peer, it presents a corrupted copy of the file, forcing a repeated request and download in order for the client to obtain the true file. Alternatively, in a "slow node attack," the malicious peer points the client to a slow or overloaded peer with the goal of increasing the client's delay. Such attacks are particularly malicious as they consume resources in both the data and control planes. Moreover, we show that false-reply attacks possess an extraordinary scaling behavior, in which the attacker can significantly degrade the performance of the entire p2p system while controlling only a small fraction of nodes.

Even a small percentage of nodes in a large-scale system can represent 100s or 1000s of hosts. We note two mechanisms by which attackers can control numerous hosts. First, the attacker can deploy all malicious nodes itself at a single or multiple Internet Data Centers.[3] A second way to launch an attack is by subverting peers via a "trojan horse" program that serves corrupted content. Trojan horse programs are already common on both the Internet (*e.g.,* those spread via email viruses, worms, and the web) as well in p2p systems [23].[4] This latter scenario could be employed by "resource-poor" malicious users who wish to deny service to others.

Modeling File Targeted Attacks. To study the resilience of p2p networks to file-targeted attacks, we develop a discrete-time model that enables us to study the spread of good and bad copies. We initially assume a fully cooperative p2p environment. We demonstrate that in this case the pollution attack has a serious scalability limitation, and is unable to prevent the spreading of good copies in the system. Without full cooperation, however, user-behavior factors, such as (i) slow and incomplete removal of corrupted copies, (ii) unwillingness to share downloaded files, and (iii) lack of persistence in downloading files when the system is under attack, prevent good copies from spreading in the system and render the attack far more effective.

Modeling p2p Network Attacks. Network-based attacks are dependent on the network topology. We model a two-level hierarchy, a k-regular graph, and a power-law graph. Two-level hierarchy topologies occur in systems with supernodes such as Gnutella and KaZaA. k-regular graphs arise in structured p2p networks such as CAN [19], Chord [19], Pastry [20], Tapestry [13], and Kademlia [17]. Finally, power-law graphs can arise in a number of ways (see [1]). In particular, they occur as a protocol objective in Freenet [9], and in networks in which the access link capacity has a heavy-tail distribution and a node's degree is made proportional to its access link capacity. We also model the effects of different client counter-DoS strategies such as random and redundant reply selection and reputation systems.

Our findings for modeling network-based attacks are as follows. First, the model characterizes how the additional protocol functions of supernodes yield significant leverage to DoS attackers that obtain supernode status (in today's Gnutella, nodes self-declare themselves as supernodes by advertising a high access link bandwidth). Second, non-hierarchical k-regular graphs incur a different scaling for resilience to attackers. The "collapse" points for such graphs typically occur only with very large path lengths (*e.g.,* greater than 10), which occur either in very large scale systems or in networks that route via long paths specifically to achieve anonymity, *e.g.,* as in [6, 9]. Third, we find that power-law graphs present an acute vulnerability to DoS in cases in which malicious nodes are able to insert themselves in the high-degree "hubs" of the graph. While vulnerability of power-law graphs to DoS attack and failure is well established, e.g., [2, 9], no prior study has explored a scenario in which highly-connected nodes *participate* in the attack.

Finally, the analytical model characterizes the impact of the client's reply-selection policy. The worst policy under attack is the "best peer policy," in which a client selects the peer advertising the best performance. Because attackers can easily falsify performance information, a victim that "believes" reported information is only successful when *no* false replies are received. Furthermore, our analytical model characterizes system performance in the presence of non-perfect reputation systems, and under various client reply-selection policies. We show that reputation systems with even extremely small inaccuracies (incorrect belief that a malicious node is non-malicious or vice versa) are unable to improve the performance of different variants of the "best peer policy."

Simulation Experiments. The key result of our simulations is the characterization of the tradeoffs between performance of the system in the absence of an attack and its resilience during an attack. Experiments confirm the extreme vulnerability of the "best peer policy." They also demonstrate that if the users instead select their download source randomly, the system becomes far more resilient (goodput decreases only linearly with the number of attackers), but at the expense of a substantial performance penalty in the absence of attacks. This tradeoff between resilience against attacks and performance in the absence of attacks is quite pronounced. For instance, for the particular parameters used in our simulation, a "best peer" strategy leads to a virtual system collapse when the attack can commandeer 2.5% of the supernodes. In contrast, for the same set of parameters, choosing a random peer from the received query responses prevents collapse even under a high number of attackers. This resilience comes at the expense, however, of a sevenfold increase in average download time in the absence of an attack.

We next present a brief background on p2p systems. In Sections 3 and 4 we present the file- and network-targeted DoS scenarios. In Section 5 we present our analytical model and in Section 6 simulations. Finally, in Section 7 we conclude.

[1] http://www.overpeer.com

[2] http://www.retsnap.info

[3] While the costs of such a cluster along with sufficient bandwidth to serve the false content could be 100s of thousands of dollars, such amounts can be quite modest in certain scenarios. For example, in the context of networks used to trade copyrighted material, the RIAA estimates $4B/year in lost revenue due to mp3 trading and spends an estimated $17M/year in legal fees.

[4] For example, reference [23] describes how many p2p users were thwarted by a spyware program bundled to feign being third-party advertising software. The application installed even if users opted not to install it.

2. BACKGROUND ON PEER-TO-PEER SYSTEMS

P2p systems can be broadly classified as structured or unstructured based on whether there is any inherent structure in the system that can be exploited to efficiently locate files.

In unstructured p2p systems such as Gnutella,[5] a given file can be stored at any node in the system. The original version of Gnutella used scoped flooding to locate a file. While this method is highly robust and flexible, it is not scalable. To address the scalability problem, newer versions of Gnutella as well as other unstructured p2p systems such as KaZaA[6] use a two-level hierarchy. The first level of the hierarchy consists of leaf nodes, and the second level consists of more powerful nodes, called supernodes. Each leaf node is connected to one or more supernodes. A supernode maintains a directory of all files stored at its leaf nodes. When a leaf node queries a file, it sends the query to its supernode. If the supernode knows the location of a file copy (i.e., if one of its leaf nodes stores the file), it sends the answer back to the requester. Otherwise, the supernode floods the query to other supernodes. Since the number of supernodes is much smaller than the total number of nodes in the system, such hierarchical p2p systems are more scalable than the original Gnutella.

Freenet [8, 9] is an unstructured p2p network whose aim is to provide anonymity and censorship resistance. Each file in Freenet is assigned a unique ID by hashing the file content. Each node maintains a routing table consisting of the IDs of the files stored locally and at the neighbor nodes. When a new file is inserted, the file is routed according to its ID and stored at all nodes along the path. Similarly, when a file is retrieved, the file is copied along the path from the source to the requester. This makes Freenet highly resistant to censorship, as it is hard if not impossible to locate all copies of a specific file. Furthermore, trying to locate a file will result in the file being copied at even more nodes.

Structured peer-to-peer networks such as CAN [19], Chord [19], Pastry [20], Tapestry [13], and Kademlia [17] partition a global ID space across all nodes in the system. As a result, each node becomes responsible for a chunk of the ID space. Each file is associated with a unique ID, for example, by hashing the file content or the file title into the ID space. A file is then stored at the node responsible for the file's ID. Alternatively, a file can be stored at an arbitrary node in the system, as long as a pointer to the file is stored at the node responsible for the file's ID. In either case, one needs to find this node in order to retrieve the file. Thus, the basic operation in a structured peer-to-peer network is: given an ID, find the node responsible for that ID. Structured p2p networks are very efficient in locating such a node. In general, they can find the node responsible for a given ID by contacting only $O(\log N)$ nodes, where N is the number of nodes in the system.

Structella [4] is a hybrid proposal based on Pastry. Like the original Gnutella, Structella uses flooding to locate files, but does so in a more efficient way. In particular, Structella uses the underlying structure of Pastry to send no more than one flood message per virtual link. This helps to reduce the flooding cost by a factor of k, where k is the average degree of a node in Pastry. In this paper, we assume that the replies are sent back to the requester using the Pastry routing protocol.

3. FILE-TARGETED DOS ATTACKS

It has been shown recently that the music industry has undertaken serious efforts to combat file sharing of copyrighted content by depositing large volumes of corrupted (polluted) files into p2p systems such as KaZaA [7, 16]. In such an attack, a malicious node advertises a corrupted file, and eventually distributes this copy if it is chosen by another peer. Unlike for network-targeted attacks, the p2p network topology does not play a role in the effectiveness of a file-targeted attack. Instead, the user-behavior factors such as willingness to share files, speediness in removing corrupted files, and persistence in downloading files under attack determine the spread of polluted files. We present a simple model to evaluate the file-sharing dynamics under this "pollution" attack.

In particular, we model the number of peers that have a good (non-corrupted) copy of a particular file, and the number of peers that have a bad (corrupted) copy of the same file. Indeed, there is evidence that the music industry protects only certain audio and video files, usually the new releases [7, 16], and thus our goal is to explore the dynamics in sharing these files. In addition, the total number of nodes considered in our system model is only a *subset* of nodes that can be present in a p2p network.

The modeling assumptions are as follows. First, upon a query for a file, the user is presented with the list of *all* nodes that advertise that particular file. Second, each node can advertise only *a single* copy of a specific file. This policy prevents a single malicious node from performing large-scale attacks against a certain file, and it can easily be enforced through the search mechanism. Finally, we assume that a user picks a random file from the list. In light of recent DoS attacks against p2p file sharing systems, this is a likely counter-DoS method, and we show in Sections 4.3 and 6.4 that this is indeed the most successful client counter-strategy among the ones that we consider.

3.1 Spreading the Pollution

While users have a clear incentive to keep a good copy on their machines, it is possible that a bad copy remains on a non-malicious user's machine for a certain amount of time. If a corrupted file is not immediately inspected and removed after the download, it remains on the machine for a certain amount of time, and during that time it can be downloaded by other users. Moreover, there is evidence that downloads in p2p file-sharing systems are often made in the background and that content is typically examined later [11].

Denote by M the total number of users that are either interested in downloading a certain file or already have a copy (either good or bad). Next, denote by g_i and b_i the respective number of peers that have good and bad copies of the file at time interval i (a time interval corresponds to one hour). Further, denote by s_i the interest-rate factor that determines how many of the interested nodes actually send a query for the file during the i-th hour. Then, the number of nodes that have a good copy of the file in hour $i+1$ is

$$g_{i+1} = g_i + (M - b_i - g_i)s_i\frac{g_i}{g_i + b_i}. \quad (1)$$

In Equation (1), $M - b_i - g_i$ is the number of nodes that are interested in obtaining the file, but still do not have a copy of it. $\frac{g_i}{g_i+b_i}$ is the probability that the users that have sent a query in the i-th hour download a good copy of the file. Next, define R_i as the number of nodes that are "infected" with a corrupted copy during the i-th hour as

$$R_i = (M - b_i - g_i)s_i\frac{b_i}{g_i + b_i}. \quad (2)$$

The above term is similar to the one from Equation (1), with the difference of the factor $\frac{b_i}{g_i+b_i}$, which is the probability that the file

[5] http://gnutella.wego.com
[6] http://www.kazaa.com

downloaded in the i-th hour is polluted. Next, denote by p_j the probability that an infected node removes a corrupted copy after j hours, and denote by L the maximum number of hours for which a corrupted copy can remain on the user's machine. Then, the number of polluted nodes in hour $i+1$ will be

$$b_{i+1} = b_i + R_i - \sum_{j=1}^{L} p_j R_{i+1-j}. \quad (3)$$

where $\sum_{j=1}^{L} p_j = 1$. Equation (3) provides a relationship between the number of polluted nodes in two consecutive hours. On one hand, the number of polluted nodes in hour $i+1$ increases by the number of nodes that get infected during the i-th hour, as defined in Equation (2). On the other hand, the number of polluted nodes in hour $i+1$ decreases by the number of peers that are "cleansed" during the i-th hour. These peers are represented by the last term of Equation (3), which sums over the fractions of peers that were infected in the past, while cleansed during the i-th hour. For example, $p_1 R_i$ is the number of peers that are both infected and cleansed during the i-th hour; $p_2 R_{i-1}$ is the number of peers infected during hour $i-1$ hour while cleansed during the i-th hour, and so on.

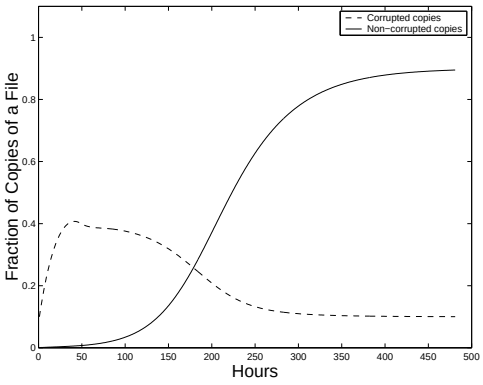

Figure 1: Spreading corrupted and non-corrupted copies

Figure 1 shows the spreading of both good and bad files in a system with $M = 15,000$ interested nodes, a large number of malicious nodes $b_0 = 1,500$, and a small number of initial good copies, $g_0 = 10$. The interest-rate factor is set to 1/24 such that each peer interested in obtaining this file attempts to download it on average once per 24 hours. Hence, not all clients instantaneously download a copy of a file, and thus the sum of the two fractions in Figure 1 is less than 1. Next, we set the parameter L to 48 such that a polluted copy can remain at most 48 hours on a user's machine. In addition, the probabilities p_i are all equal such that the lifetime of infected machines is uniformly distributed between one and 48 hours. The fraction of polluted copies in this scenario monotonically increases up to the maximum lifetime of infection, because the "infection" parameter R_i of Equation (3) is larger than the "cleansing" parameter $\sum_{j=1}^{L} p_j R_{i+1-j}$ of the same equation. The relationship between the two factors changes after 48 hours, when the number of polluted copies decreases. Furthermore, good copies spread significantly slower at the beginning because the probability to download a good copy ($\frac{g_i}{bi+gi}$) is initially very low. As time evolves, the number of good copies increases, and so does the probability to download a good copy. Eventually, all non-malicious clients (90% of all clients interested in hosting this file) manage to download a good copy of the file. At this point, $M - b_i - g_i$ of Equation (1) becomes zero, and the system reaches steady state.

Measurements from [7, 16] show, however, that the ratio of polluted to non-polluted copies in the KaZaA network remains relatively constant over time, and that good copies do *not* manage to spread. We analyze in detail below how this behavior can come about.

3.2 Cooperation and Persistence

There are two fundamental reasons that prevent files targeted by the pollution attack from spreading in the network. First, not all peers are willing to share the files that they download. Second, a user's interest for downloading newly released audio/video files quickly decreases [11]. Next, we demonstrate how both of these effects can significantly improve the success of the pollution attack.

A previous study has shown that p2p users in general are greedy, *i.e.*, most users consume data, but provide little in return [22]. This behavior is even more prevalent due to recent legal actions against p2p systems (*e.g.*, against KaZaA [14]). Denote by p_s the probability that a user is *not* willing to share a good copy of a file once it has downloaded it, and denote by g_i^P (P stands for *public* copies) the number of users at time step i that are willing to share a good copy. Then, the number of good public copies increases as

$$g_{i+1}^P = g_i^P + (M - b_i - \frac{g_i^P}{1-p_s}) s_i \frac{g_i^P}{g_i^P + b_i}(1-p_s). \quad (4)$$

Equation (4) is similar to Equation (1), with the difference that the total number of copies at time interval i is expressed as a function of good public copies ($g_i = \frac{g_i^P}{1-p_s}$). Also, the increase in the number of good public copies is reduced by the factor $(1-p_s)$, as compared to Equation (1). It can be shown that equations similar to Equations (2) and (3) govern the spreading of polluted copies.

In addition to clients being unwilling to share files, the actual interest (request) rate for a particular file influences the spreading of both good and bad copies. A measurement study [11] indicates that the interest rate for new popular objects (those typically targeted by the pollution attack) tends to decrease significantly after only a few weeks. While no study explicitly measures the user behavior in the presence of a file-targeted attack, it is reasonable to assume that the interest rate for a certain file decreases even faster under a pollution attack, because users become frustrated after downloading bogus copies. Here, we evaluate a simple linear interest-rate function $s_i = 1/24(1 - \frac{i}{24}0.15)$. This means that, on average, 15% of users give up after the first day, another 15% after the second day, and so on. While not representative of an actual scenario, our main goal here is to illustrate the impact of users' persistence on the effectiveness of the attack.

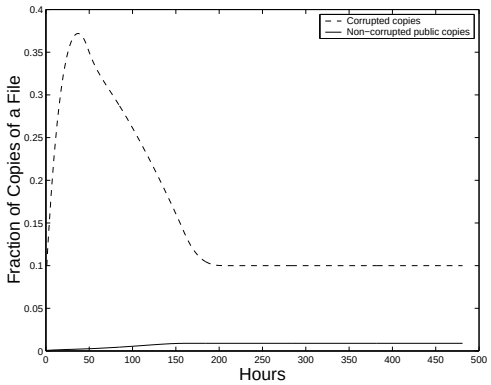

Figure 2: The impact of users' greediness and persistence

Figure 2 depicts the effects of user greediness and persistence on the rate at which good copies spread. All parameters are the same as in the previous example. In addition, we set the probability that a user is willing to share the file to 0.6 ($1 - p_s = 0.6$), while the interest-rate is modeled with the linear function above. First, the decline of the number of bad copies in Figure 2 is more sharp than the decline in Figure 1, which is due to the low persistence level. After realizing that they have downloaded a polluted copy of a file, users may give up and make no further attempts to download the file. Hence, the probability to get re-infected decreases, and so does the number of infected nodes. The key point, however, is that if users get discouraged quickly, *good* copies are never successfully distributed in the network. In our scenario, the interest rate factor s_i of Equation (4) converges to zero approximately beyond 180 hours, forcing the system to reach a quite unsatisfactory steady state.[7] Interestingly, measurements from KaZaA [7, 16] show that the number of good and bad copies for newly released files does not change much over time, indicating that the network operates in a "depressed" mode (like the one in Figure 2 beyond 180 hours), in which clients do not manage to increase the number of good copies.

In summary, the main reason for the success of file-targeted attacks applied in today's p2p systems is the user behavior. In particular, the key factors are (i) negligence in cleansing the machines infected by polluted copies, (ii) users' unwillingness to share downloaded files, and (iii) a low persistence level. However, such an attack is unable to prevent the spreading of good copies in a fully cooperative p2p environments (that do not exhibit the above (i)-(iii) behavior) with a sufficient interest rate for a certain file, as indicated in Figure 1.

Thus, in the rest of the paper, we anticipate the next step in the "arms race" between the attackers and defenders, and treat a class of more sophisticated DoS and counter-DoS strategies.

4. NETWORK-TARGETED DOS ATTACKS

We present a class of DoS attacks targeted against entire p2p networks. The key differences between such attacks and the file-targeted (pollution) attack are as follows. First, in network-targeted attacks, an attacker responds to *all* queries, whereas in the pollution attack it only replies to queries for a set of targeted files that are being protected. Second, in network-targeted attacks, the attacker is able to *intercept* a query for a downstream node and falsify the reply on the reverse path. Hence, a query that follows a path with even a single malicious node gets a response pointing to a bogus file.

4.1 System Model

We consider a p2p file sharing system in which the interaction between the clients and system occurs in two steps:

Query. The client queries the system for a particular file, and the system returns a number of replies. Each reply contains the location of a copy of the queried file, and information about the node storing the copy. Without loss of generality, we assume that a reply contains (1) the IP address of the node storing a copy of the queried file, and (2) sufficient information for the client to calculate the estimated time to download the file from this node, e.g., the node's queue length (ideally including file sizes), the maximum number of simultaneous uploads, and the access link bandwidth.

Download. The client selects a node among the nodes contained in the replies it has received, and contacts that node to download the file. Clients may employ a number of selection policies as described below.

In the rest of the paper, we refer to the first phase of the interaction as the control plane, and the second phase as the data plane.

[7]While we do not consider client's arrival and departure dynamics, shorter lifetimes of nodes can further slow down spread of good copies.

4.2 Attacker Strategy

An attacker can interfere with both the control plane and the data plane. In this section and in the rest of the paper we consider the following scenario: Upon receiving any query, a malicious node forwards it normally. Upon being requested to forward any reply, however, the malicious node modifies the reply with false information. We consider two cases:

False reply attack. The attacker falsifies the reply by replacing the replying peer's identity with its own and by advertising a very low expected transfer delay. This strategy allows the attacker to respond to requests for files for which it has no or limited information (*e.g.*, the attacker does not know the exact file name). If selected by the client, the node transfers a corrupted file.

Slow node attack. The attacker points the client to a non-malicious but low-bandwidth peer, and lies about that peer's capabilities, *i.e.*, it changes the advertised delay of slow nodes. It also drops replies from fast nodes.

In both cases, we assume that the attacker cannot respond to queries directly, but rather must wait for legitimate replies from downstream in order to modify them. This is because, typically, queries to p2p networks are not very precise, *i.e.*, they result in multiple files in the result set, which is ultimately filtered by the user when making the final download decision. We consider that attackers cannot modify the query forwarding algorithm executed by a legitimate node. Thus, a query that follows a path consisting only of legitimate nodes always generates a correct reply.

The space of possible attacks in a p2p systems is immense. We focus on a limited class of attacks that aim to attack system-wide performance. Even within this class of attacks, we do not consider all possibilities. For example, we do not explore attacks on the routing protocol, that are treated elsewhere [5].

4.3 Client Strategy

In response to a query, a client receives a set of replies pointing to different nodes. The main decision that the client needs to make is which one of these nodes to ask for a copy of the file. We consider the following selection strategies:

Best. The client selects the node that advertises the best performance, *i.e.*, the node with the lowest estimated delay (the node's queue length times the file size times the maximum number of simultaneous uploads divided by the access link bandwidth).

Random. The client selects a random node, independent of the nodes' advertised resources.

Redundant best. The client performs redundant downloads from the C nodes with the lowest estimated delay. Once the first download finishes and the content is verified for correctness, the other downloads are stopped.

Redundant random. The client performs redundant downloads from C peers, but chooses those C peers randomly.

File Chunking. The file is sliced into P chunks, and the client downloads a chunk from each of P different peers in parallel. File chunking is already used in today's systems to improve response time for downloading large files. Selection of these peers can be *best* or *random*.

Reputation Systems. We consider a simplified model in which a reputation system is employed to mark peers as malicious or non-malicious. We do not attempt to model the specifics of the protocol beyond the fact that it is imperfect, i.e., it has a non-zero false-

negative and false-positive probability. This abstraction enables us to evaluate how accurate the reputation system must be in order for the system to be resilient to DoS attack. We do not attempt to study key challenges for reputation systems such as assurance of persistent identity, prevention of collusion for false accusation or false praise, binge bad behavior after good behavior, etc. [10, 12, 15, 18].

Detection. For the download of a complete file, we assume that the client can detect whether a file is corrupted only after it has downloaded the entire file. The client then selects a different peer from the response list and downloads the file again. For file chunking, we consider two possibilities. First, as an upper bound on performance, we consider the case in which the client can detect a corrupt chunk as soon as it receives it, and immediately downloads that chunk from an alternate node from the set of nodes that had replied to the query. Second, as a lower bound on performance, we consider the case in which the client must first download all chunks before inferring that the file is corrupt. At this time, the client is not able to infer which chunk is corrupt, only that the file is corrupt. Subsequently, the client downloads all chunks from new peers from the set of replies to the original query, and we evaluate this approach later in simulations.

Finally, like the space of attacks, the space of possible defenses is also quite large, and our scope is limited to the above strategies. Despite these limitations, our study provides a key step towards understanding and quantifying the vulnerability of p2p systems against network-targeted attacks.

5. MODELING RESILIENCE TO NETWORK-TARGETED ATTACKS

We develop simple models to evaluate the impact of a collection of DoS nodes on p2p system performance focusing on three issues: hierarchy via supernodes, k-regular topologies and path length, and power-law graphs.

5.1 Supernodes and Hierarchy

Our objective here is to develop a model that isolates the impact of malicious supernodes on a system's DoS resilience. In particular, supernodes have increased control plane functions that can be exploited by an attacker with the following properties relevant here: (1) requests and replies are routed via an inter-connected mesh of supernodes, and (2) supernodes reply to queries on behalf of their leaf nodes. Consequently, a malicious supernode can exploit all of these properties to more successfully spread false information in the false reply attack described in Section 4.

Denote the number of peers in the system by N, the number of supernodes by S, and the number of malicious supernodes by s, with $s \leq S \leq N$. Moreover, to provide a lower bound on the damage of the attack, we consider fully replicated content in which all nodes store all content. This maximizes the number of "true replies" to a query. Consequently, in this scenario each query results in N responses and a particular response is *false* if the reply has been generated or forwarded by a malicious supernode.

Consider a graph in which each supernode is equally likely to be chosen for each hop, and the path length has H supernode hops, where H is a random variable. A response is valid only if all H nodes visited are not malicious so that

$$P(false|H=h) = \frac{s}{S} + \left(1-(1-\frac{s}{S})^h\right)(1-\frac{s}{S}). \quad (5)$$

and $P(false) = \sum_h P(false|H=h)P(H=h)$. The first term is the probability that a peer is directly connected to a malicious supernode (and hence all its requests fail) and the second term is the probability that the request fails given that the peer is not directly attached to a malicious supernode times the probability that a node is not directly attached to a malicious supernode.

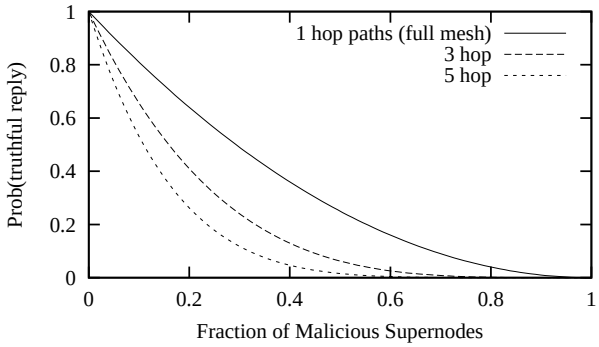

Figure 3: The Role of DoS Supernodes

Figure 3 depicts the probability of receiving a true reply as a function of the fraction of malicious supernodes s/S and for a constant H. Thus, with 0 attackers, 100% of replies are truthful, whereas with 10% of malicious supernodes the probability is reduced to 81% for $h = 1$, which represents a fully interconnected mesh of supernodes such that all paths are one hop. For longer paths and $h = 3$, the probability of receiving a truthful reply for 10% malicious supernodes is reduced to 65.6% and for $h = 5$, to 53.1%. Thus, the attack is increasingly powerful with larger h as DoS nodes have increased opportunity to intercept queries.

An example scenario with a ratio of supernodes to non-supernodes of 10, a path length of $h = 4$, and a network size of N =100,000 peers, s =1,000 attacking supernodes (10% of supernodes) produces a truthful reply probability of 59%. Thus, such an attack indeed has a "multiplier effect" in which 1% of bad nodes reduces truthful replies by 41%. While this example attack may appear to be relatively mild at first glance, we show in Section 6 that the "positive feedback" of repeated retransmission induced by such false replies can indeed have a significant effect on successful file transfer delay and system goodput.

5.2 k-Regular Topologies and Path Length

Our goal in this section is two fold. First, we aim to model structured peer-to-peer networks such as CAN [19], Chord [19], Pastry [20], and Tapestry [13], whose underlying topology can be approximated by a k-regular graph, where k is usually $O(\log N)$. Second, we want to explore the effects of the path length on the robustness of such networks. Typically, the length of the path in these systems is $O(\log N)$, but it can be significantly larger when users desire anonymity. Indeed, as described in Section 2 and reference [6], anonymous communication inherently requires high hop counts in the absence of a trusted third party anonymization service.

Let n be the number of malicious nodes, and H a random variable denoting the number of hops on the path. Under these assumptions and with each node being equally likely to be on the search path, failure occurs if *any* node along the path is malicious such that

$$P(false) = \sum_h \left(1-(1-\frac{n}{N})^h\right) P(H=h). \quad (6)$$

In examples from Freenet with a 100,000 peer network, H has mean 10, and first and third quartiles of 3 and 40 [8]. For Figure 4 we consider H to be constant, taking on values of 3, 10, or 40. The figure and Equation (6) clearly indicate that such high hop

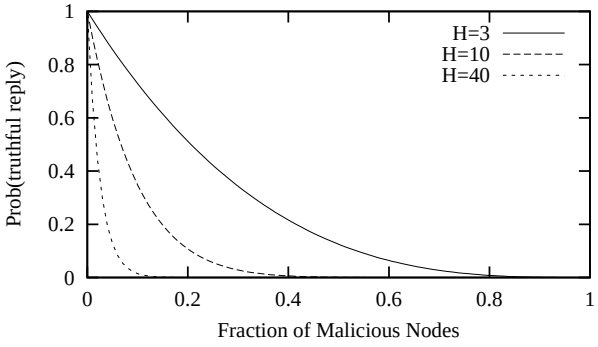

Figure 4: High Path Length

counts provide strong leverage for attackers, even though attackers no longer have the leverage of being a supernode in this scenario. For example, an attacker with 10% of nodes can reduce the truthful-reply probability to nearly 0 when h is 40. In comparison with the supernode case in which 1% of *total* nodes are malicious, with a flat network structure, even a high hop count of $h = 10$ reduces the truthful-reply probability quite mildly to 90%.

5.3 Power Law Topologies

Above we considered graphs in which each node is equally likely to be on the path of a query response. However, attacks can be far more devastating for graphs with power law structure if malicious nodes are able to insert themselves into the highly connected "hubs" of the graph. Given that the existence of power law graphs in p2p networks has been previously established (*e.g.*, reference [8]), our objective here is to explore the extent to which such topologies impact a system's DoS resilience. While fault tolerance and resilience to *external* DoS attacks has been studied for power law graphs in [2, 9], here, we consider highly connected nodes to be *participating* in the attack.

To study this effect, consider a network consisting of N nodes, where node i has degree d_i. Then we have that for random lookup operations, the expected number of lookups that traverse node i is at most proportional to its degree d_i. This observation is justified as follows: Consider a structured p2p network like Chord, Pastry, or Tapestry. Let G be the graph representing the network topology. Without loss of generality assume that nodes are ranked by their degree where node 1 has the highest degree and node N has the lowest degree. Further assume that each node covers a range of ID space proportional to its degree. Construct a new graph G' as follows. Replace each node i in G with $n_i = d_i/d_N$ virtual nodes. Then each virtual node in G' routes approximately the same number of lookups. By this argument, node i in the original graph G will route a number of lookups proportional to the number of its virtual nodes, *i.e.*, d_i/d_N.

Note that the model is an approximation in that a node with high degree will actually route less than its share since a lookup in G' may traverse virtual nodes belonging to the same node in G, in which case the corresponding node in G will be counted multiple times. Moreover, if each node in G covers the same ID space then a node will route even less than its fair share of lookups.

Continuing with the model, we consider the case that the degrees of nodes in G have a Pareto distribution,

$$P[X > x] = k \times x^{-a}, \tag{7}$$

where $P[X > x]$ represents the number of nodes with degree greater or equal to x, a is the shape parameter, and k is a constant. Then the degree d_r of the nodes with the highest rank r is $d_r = k_1 \times r^{-1/a}$, where $k_1 = k^{1/a}$. Moreover, the sum of the degrees of the highest ranked f nodes is then $D(f) = k_1 \sum_1^{x=f} x^{-1/a} = \frac{k_1 a}{a-1} \times \left(f^{\frac{a-1}{a}} - 1\right)$.

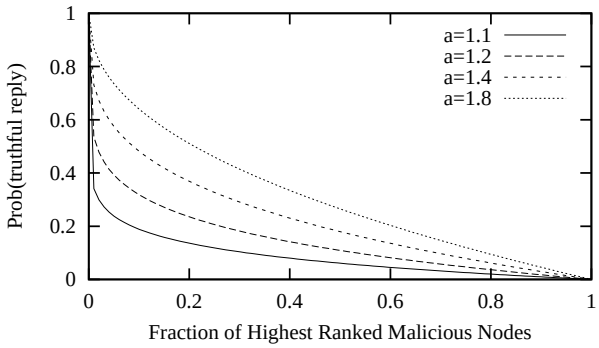

Figure 5: DoS in Power-law Networks

Thus, in contrast to Sections 5.1 and 5.2 in which each node (malicious or not) is equally likely to be a hop on the query path, here the node degree weights the likelihood of a node being on the path according to its degree. Moreover, we consider the borderline scenario for the attacker in which malicious peers are placed as the highest ranked f nodes in the graph. Then for $h = 1$ (a fully connected mesh) the expected fraction of lookups that will be compromised is bounded above by $P(false) = \frac{D(f)}{D(N)} = \frac{f^{(a-1)/a}-1}{N^{(a-1)/a}-1}$.

Figure 5 depicts numerical results for this case and a 10,000 node network and indicates that compared to the "$h = 1$" curve in Figure 3, the attack is far more severe. Most notably, all curves drop sharply with even a small percentage of attacking nodes, as even the first malicious node is the most connected node and has substantial opportunity to spread false information. The extent to which the attack scales is a function of the Pareto shape parameter with a larger a indicating a heavier-tailed node degree and a more severe attack.

Of course, in practice, queries traverse multiple hops so that the performance under $h > 1$ is most relevant. Here we consider a flat node structure (no supernodes) as in Equation (6) and again assume that a requests visit exactly h hops and that the probability to visit a node is proportional to its degree. With the highest ranked f nodes being malicious, the probability of false information is given by

$$P(false) = 1 - \left(1 - \frac{D(f)}{D(N)}\right)^h = 1 - \left(\frac{N^{(a-1)/a} - f^{(a-1)/a}}{N^{(a-1)/a} - 1}\right)^h. \tag{8}$$

Unfortunately, Equation (8) indicates that a 10,000 node network with 4-hop paths obtains devastating performance even under a modest number of attacking nodes. For example, for $a = 1.4$ and 1% malicious nodes the truthful-reply rate is only 0.38%.

Finally, note that if the joint effect of power law graphs together with high path lengths for anonymity or supernodes for scalability would make the system even more vulnerable to attack as indicated by Equations (5), (6), and (8).

5.4 Client Strategies

We next explore client counter-DoS strategies that play a crucial role in relating the probability of receiving false vs. true information to the probability of a failed vs. successful download (denoted by $P(fail)$ and $P(succ) = 1 - P(fail)$, respectively).

5.4.1 Success Under False Information

We consider the same system model in which the attacker returns bogus replies. Out of N replies to a query, the p2p user chooses to download a single file, or multiple files simultaneously, depending on the policy described below. Upon downloading a file (either good or bad), the user sends the query for another file. In other words, we assume independence between successive queries.

Select Perfectly. At one extreme, if the victim was able to know which replies are false via omniscience, then a download fails only if all received information is false. Thus, $P(succ) = 1-P(false)^N$ rendering $P(succ)$ quite close to *one* for large system sizes. However, as such a policy is infeasible in practice, we consider more realistic policies as follows.

Select "Best". A trusting user will select the "best" reply according to criteria such as advertised link bandwidth or expected download time. Unfortunately, this policy is at the other end of the extreme for yielding success as attackers will falsify such information. Consequently, the success probability in this case is given by

$$P(succ) = (1 - P(false))^N, \quad (9)$$

quite close to 0 for large system sizes. Equation (9) indicates that the download is going to be successful only if all replies to a query are correct. Otherwise, if at least one is bogus, that one is selected, and causes an unsuccessful download.

Select Randomly. When users are aware that the system is under attack, they are less trusting of advertised performance measures. If they consequently select randomly among the replies, then we simply have

$$P(succ) = 1 - P(false). \quad (10)$$

Select Redundantly. If users download C redundant copies in order to protect against false information, then the probability of successful download is

$$P(succ) = 1 - P(false)^C. \quad (11)$$

Select Best Redundantly. When users select the C "best" advertised download times, the probability of successful download becomes

$$P(succ) = \sum_{i=0}^{C-1} (1 - P(false))^{N-i} P(false)^i. \quad (12)$$

Equation (12) indicates that in the "best redundant" scenario, the download is going to be successful only if there exists at least one truthful reply within the top C replies.

File chunking. For file chunking, the above expressions directly apply to each *chunk*, assuming that the peer for each chunk is chosen independently. Denoting $P(succ')$ as the probability of successful download for a chunk, as computed above, the probability of successful download for the entire file becomes $P(succ) = P(succ')^P$ for a file sliced into P chunks. Thus, without attackers, file chunking improves performance as it increases download throughput. Yet under attack, this improvement is countered by a reduction in the probability of a successful download.

Figure 6 depicts the impact of reply-selection policies and reputation systems on the successful-download probability as a function of the false-reply probability. We set $N = 1500$. For the time being, we focus on the results for the network *without* a reputation system, *i.e.*, the "thin" curves of Figure 6. On one hand,

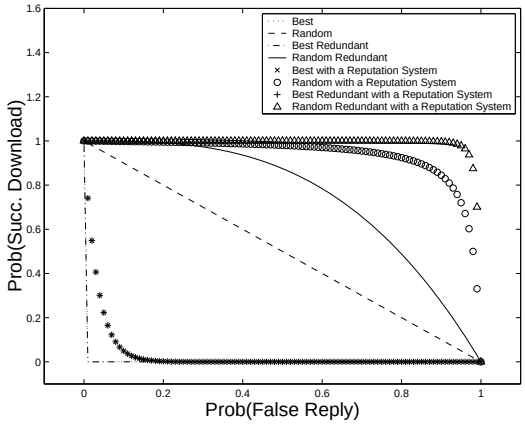

Figure 6: Reply-Selection Policies and Reputation Systems

the "best" policies (*e.g.*, Equations (9) and (12)) are highly vulnerable to very small false-reply probabilities. Indeed, if a user always chooses to download one or more files with best-advertised download times, then a small percentage of nodes with maliciously-advertised download times is enough to decrease the successful download probability to *zero*. On the other hand, a simple random strategy significantly improves the successful download probability, and the "random redundant" strategy is even more successful. We demonstrate in Section 6 that, unfortunately, random strategies considerably degrade the system performance in the absence of an attack.

5.4.2 Reputation Systems

We model the impact of reputation systems on the relationship between the successful-download and the false-reply probabilities. We do not make any assumptions about the particular reputation mechanism, since that is beyond the scope of this paper. We do, however, gauge the impact of the *accuracy* of a potential reputation algorithm. Denote f_N and f_P as the false-negative and false-positive probabilities of a reputation system. The false-negative probability is defined as the fraction of malicious nodes that are left undetected by the reputation system, while the false-positive probability is the fraction of non-malicious users that are falsely declared malicious.

After receiving N replies for a single query, the user discards all replies that are declared "malicious." For a given false-negative probability f_N, the number of correctly detected malicious replies becomes $N(1-f_N)P(false)$, while the number of falsely-detected non-malicious replies becomes $Nf_P(1 - P(false))$. Since both of the above two classes of replies are discarded by the reputation system, the "effective" number of *non-discarded* replies, N_R, becomes

$$N_R = N(1 - (1 - f_N)P(false) - f_P(1 - P(false))). \quad (13)$$

Not all of the remaining N_R replies are necessarily good. Reputation systems fail to detect malicious nodes with probability f_N. Hence, it can be shown that the false-reply probability under the reputation system, $P_R(false)$, becomes

$$P_R(false) = \frac{f_N P(false)}{1 - (1 - f_N)P(false) - f_P(1 - P(false))}, \quad (14)$$

where $P(false)$ denotes the corresponding false-reply probability

in the absence of a reputation system. Finally, by replacing N and $P(false)$ in Equations (9)-(12) with N_R and $P_R(false)$ as computed above, we obtain the successful-download probability for a given accuracy of a reputation algorithm.

Figure 6 shows the impact of a reputation system on the successful-download probability with $f_N = f_P = 0.02$. The other parameters are the same as in the previous subsection. Even with extremely small false-detection probabilities, reputation systems are unable to improve the performance of the "best" strategies. In essence, if a user always chooses to download files with the best-advertised download times, then even a small fraction of malicious nodes that manage to "survive" the reputation system's filter are able to quickly degrade the successful-download probability to zero. On the contrary, an efficient reputation algorithm further improves the "random" strategies. Again, we demonstrate in Section 6 that such strategies (even when combined with reputation systems) considerably degrade the system performance in the absence of an attack.

6. SIMULATION STUDY

We present an extensive set of simulation experiments to explore the key system factors that influence DoS resilience of p2p file sharing systems.

6.1 Simulation Preliminaries

We implemented a discrete event simulator of a p2p file sharing network with the following capabilities: (1) p2p network overlay maintenance, (2) query request and reply routing, (3) network model, (4) content distribution model, (5) search query and response processing at each node, (6) file transmission and reception, (7) user model for download selection and initiation, (8) handling queuing and rejection of file download requests, (9) multiple DoS attacker behaviors, and (10) multiple counter-DoS strategies. We elaborate on some of these factors below.

We investigated both structured and unstructured p2p overlays. For the unstructured overlay we have implemented a Gnutella network simulator, largely based on gnutellasim from limewire.org. Requests are flooded over Gnutella's overlay network, while replies are routed back to the requester along the reverse path. For the structured overlay we used FreePastry to implement the query broadcast facility of Structella, as described in [4]. The replies are sent using Pastry's usual point to point routing mechanism.

We do not model the network core and consider a scenario in which the bandwidth bottlenecks are at client access links. As such, we divide peers into high and low bandwidth peers, which in Gnutella become supernodes and leaf nodes respectively. Unless otherwise specified, access link rates are uniformly distributed between 56 kb/sec and 1 Mb/sec for the leaf nodes, and 1 Mb/sec to 10 Mb/sec for the super nodes. We use a Zipf distribution to represent file popularity and file replication. Moreover, queries are processed at each node with the non-malicious node's reply probability taken from a Zipf distribution of file replication, its rank having been given in the query request. We make the simplifying assumption that the file popularity distribution is the same as the replication distribution.

Finally, file transmission is simulated by allocating bandwidth according to access link speeds being max-min fairly shared among all downloads at an endpoint. Unless otherwise indicated, the number of nodes in the system is set to 10,000. In most cases, our key performance measures are the *probability of truthful reply* and the *average system goodput* (rate of successful transfer of true content) normalized to the number of non-malicious users. Each scenario is simulated 10 times and we report averages. Due to low variance in the output, confidence intervals are shown only in Figure 8.

6.2 Baseline Experiments

We first consider a baseline scenario for the two classes of network-targeted attacks described in Section 4: the false-reply and the slow node attacks. The scenario has a ratio of supernodes to non-supernodes of 1:10. We assume that the attacker has no limit on the number of simultaneous uploads, has high bandwidth, and responds to queries with predictions of low delay. Clients implement no counter-DoS strategies and select peers one-at-a-time based on their reported expected delay. All other parameters are set as described above.

6.2.1 *False Reply Attack*

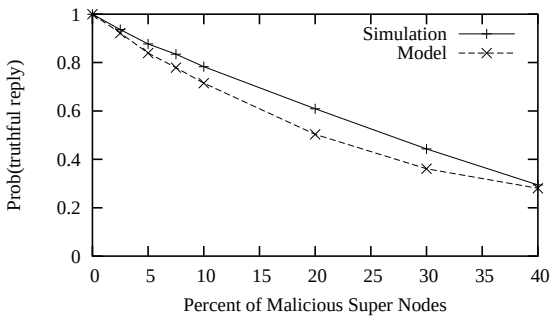

Figure 7: Gnutella Fraction of Truthful Query Replies

Figure 7 shows the probability of a node receiving a truthful reply for a Gnutella overlay with h set to the mean reply-path length, and a TTL of 3. Note the correspondence in scaling behavior between the simulation and the model. For example, with 10% malicious supernodes, the simulations measure 65% probability whereas the model predicts 75%.

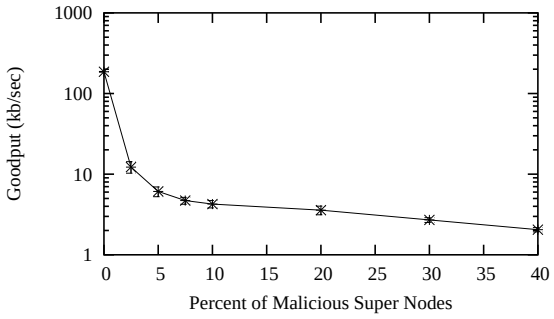

Figure 8: Baseline Attack

Figure 8 depicts the effects that the attack has on system goodput and indicates its tremendous scaling behavior characterized by two regions. Curve fitting indicates an excellent match with a 2-stage hyperexponential. The fast initial drop (indicated by the first region and the first exponent) shows that even a small number of malicious supernodes, only 0.25% of all nodes, causes the system goodput to nearly collapse. Surprisingly, the corresponding truthful reply probability in this scenario is as high as 95%. Even with such a small percentage of false replies, the probability that the set of replies to a query contains *at least one* false reply is quite high. Because the malicious nodes advertise lower expected delays than non-malicious nodes, these false replies are very likely to get chosen. In addition, the choice of a malicious peer results in a failed download, and one or more retries, creating a "positive

feedback" loop that increases load and reduces goodput. Goodput does not drop all the way to zero (indicated by the second region and the second exponent), because there are queries for which the user waits only long enough to receive replies from nearby nodes, which are not malicious in every neighborhood. Finally, while we do not show users' delay results due to space constraints, they are similar to goodput trends. For example, with 2.5% of malicious supernodes, the average users' perceived delay increases by more than an order of magnitude.[8]

6.2.2 Slow Node Attack

Our results (figures not shown due to space constraints) indicate that the slow node attack has marginal effectiveness on goodput. While one might expect (as indeed the authors previously did) that this attack would be effective as fast supernodes would remain unused while dial-up lines would become overwhelmed, the system remains far more resilient. The key reason is that this attack lacks the "positive feedback" of the false-reply attack. A false reply results in a failed download, which requires potentially repeated retries increasing delay and load. In contrast, a slow-node reply only results in a single download, albeit from a slow node. A secondary factor is that while the slow node attack reduces the utilization of non-malicious high-bandwidth supernodes, when such nodes do transmit a file, which happens with fairly high probability, the delay is quite low given their low queue length and high available bandwidth.

Consequently, an important finding is that a successful network-targeted attack requires system resources (bandwidth and storage) vs. only transmitting false information (*i.e.*, redirecting peers to the slowest peer). Thus, attackers must either (i) invest significantly in their own infrastructure or (ii) exploit software vulnerabilities in order to commandeer the resources of otherwise non-malicious peers.

6.3 System Factors

6.3.1 Overlay Structure and Hierarchy

We simulate the baseline DoS attack on systems using both a two-level hierarchy of Gnutella and the Pastry-derived Structella overlay networks. Figure 9 shows that the probability of receiving a truthful reply under attack is substantially higher when using Structella, even though the average path lengths are approximately the same (equal to 3). In the Structella scenario, approximately 5% of the nodes must be malicious in order to degrade the probability of truthful reply to 0.95. This is approximately 20 times the percentage of malicious nodes needed in the Gnutella scenario to create the same effect.

Both hierarchical and structured p2p networks aim to solve the scalability problem by making flooding much more efficient. While both schemes manage to do so, the two-level hierarchical approach is far more vulnerable to DoS attacks. In a two-level hierarchy an attacker can strategically position malicious nodes as supernodes (as we did in our experiment). Requests and replies are routed via supernodes, and supernodes reply to queries on behalf of their leaf nodes, significantly increasing the probability that a query traverses a malicious node. On the contrary, Structella is far more resilient to DoS attacks because such strategic positioning of malicious nodes is not possible with structured p2p networks that lack hierarchy.

In addition, Figure 10 depicts the system goodput as a function of the fraction of malicious nodes. The goodput collapse point moves from 0.25% of the nodes with Gnutella to approximately 5% with

[8]Henceforth we depict linear scales as some scenarios result in linear scaling.

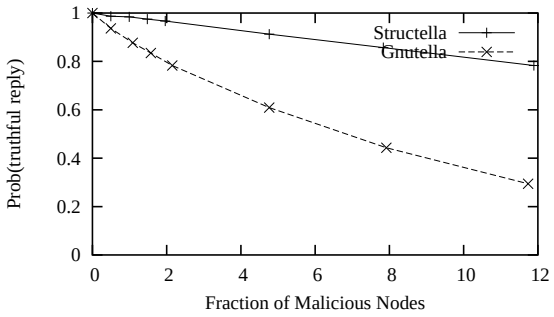

Figure 9: Overlay Structure and Hierarchy

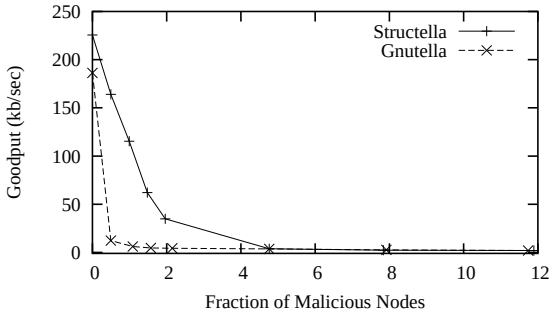

Figure 10: Overlay Structure and Hierarchy

Structella. As discussed above, both of the points correspond to a false reply probability of 0.95, which in this scenario is enough to collapse the system goodput.

6.3.2 Path Length

In Section 5 we showed that, independent of the graph structure, vulnerability to DoS increases with increasing path length. Below, we quantify the percentage of malicious nodes capable of collapsing the network goodput as a function of the path length.

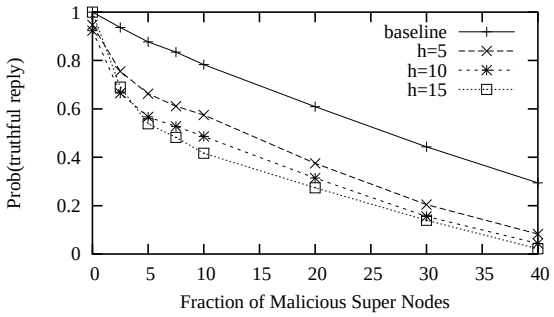

Figure 11: Probability of Truthful Replies for Long Paths

Figure 11 shows the system degradation with increased path length. While in the baseline scenario the attacker needs to control 2.5% of supernodes in order to degrade the truthful reply probability to 95%, this percentage significantly decreases with increased path length. For example, when the average path length is 5 instead of 3, an attacker needs to control *less* than 1% of supernodes in order to collapse the system goodput. Thus, while longer paths do foster anonymous communication, they significantly increase a system's vulnerability to DoS attacks.

6.4 Victim Counter Strategies

P2p users do not sit by idly when the system is under attack. They use trial and error to find effective counter-DoS strategies to improve their performance. Such users may invoke multiple downloads in order to decrease their own delay, perhaps without consideration of adverse effects on others' performance. Consequently, we consider a number of parallel download and randomization techniques. In addition, we evaluate to what extent a reputation system can improve the system resiliency to DoS attacks.

6.4.1 Best Redundant Download

We first consider parallel downloads of a file from the set of N best advertised files.

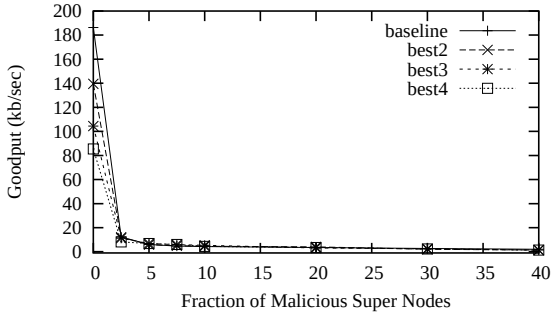

Figure 12: Best Redundant Download

Figure 12 indicates that the best redundant strategy offers no significant resilience against attack. The "N-best" strategy is still significantly thwarted by false information. Indeed, if a user always chooses to download one or more copies of a file with the best-advertised download times, then even a small percentage of malicious nodes is enough to degrade the system performance. The key insight from the figure is that the above scheme introduces a substantial goodput penalty for transferring multiple copies of files in parallel, even under no attack, due to wasted resources until the first transfer completes.

6.4.2 Random Redundant Downloads

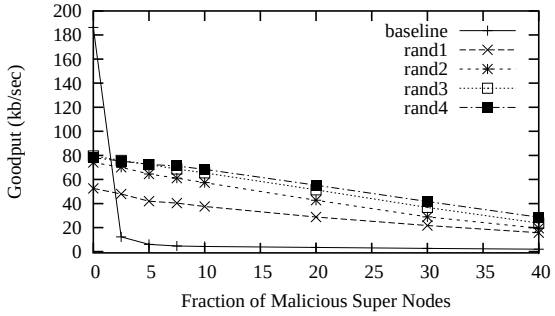

Figure 13: Random Single and Redundant Downloads

On one hand, when clients select randomly among replies, the false resource information supplied by attackers is ignored. Hence, attackers cannot attract clients by claiming to have low queues or high-speed access links. On the other hand, randomization implies that clients must ignore performance-related information attached to query replies forcing them to select a less-than-optimal choice, even if avoiding the attacker. Moreover, randomization does not preclude a client from selecting a malicious peer, it merely decreases the probability of it occurring.

These combined effects are illustrated in Figure 13. Consider first the case of a single random selection (the curve labeled *rand1*). Without attack, random selection results in a 72% decrease in goodput compared to the best-peer selection policy in the absence of attack. The attack scales, however, quite poorly, without the sharp knee that characterizes the baseline attack. Redundant download with a small redundancy factor increases the goodput, because it provides better protection against false information. With larger redundancy factors goodput decreases, because of the extra load inflicted on the system by the increasing number of redundant transfers.

6.4.3 Reputation Systems

Finally, we evaluate the impact of the accuracy of a reputation mechanism, focusing on the false-negative probability (fraction of malicious nodes undetected by the reputation system). This is because our model (*e.g.,* Equation (14)) predicts that this probability dominantly impacts the resilience to DoS attacks in the presence of reputation systems.

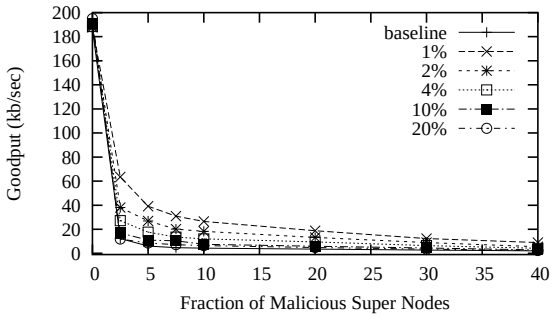

Figure 14: Reputation System and Best Selection Policy

Figure 14 shows the system goodput in a scenario in which the users apply the best selection policy, while the false-negative probability varies from 1% to 20%. Indeed, in the presence of reputation systems, the clients might feel confident to download files with the best advertised delay. The shape of the curves in the presence of the reputation system in Figure 14 is quite similar to the baseline curve where no reputation system is applied. Unlike other client counter measures, here the best system performance is retained in the absence of an attack. However, in the presence of malicious nodes, the system performance significantly degrades. While the performance is not as poor as in the baseline scenario, it is far from ideal. For example, when the percentage of malicious supernodes is as small as 2.5%, and the false-negative probability of the reputation system is only 1%, system goodput degrades to about 32% when compared to the no-attack case. If a user always downloads files with the best advertised times, then a small number of malicious nodes can degrade system performance, even when the reputation system is highly accurate.[9] As the percentage of malicious nodes increases, the positive effects of the reputation system start to fade, as predicted by our model.

Thus, given that reputation systems alone are insufficient to isolate the network from the attack, clients may start applying randomization as another level of protection. Figure 15 shows the com-

[9]While it may appear possible to build a *perfect* reputation system, the malicious nodes can apply many counter-measures (e.g., often change identity or occasionally upload a non-polluted copy of a file) to keep the false-negatives and positives non-zero.

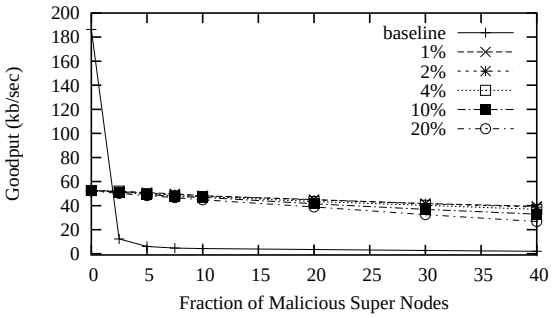

Figure 15: Reputation System and Random Selection Policy

bined effects of the two counter-DoS strategies. While the goodput performance under attack is indeed improved, it is still far below the best achievable goodput, which is 185 kb/s in this scenario. Moreover, due to randomization, system performance is inevitably degraded in the *absence* of an attack.

7. CONCLUSIONS

We analyzed DoS attacks against both popular files and entire p2p file sharing systems. We produced an extensive set of analytical models and simulations, and our findings are as follows. (i) File-targeted (pollution) attacks applied in today's p2p networks are largely inefficient in cooperative p2p environments due to scalability limitations; the main reasons for their current success are that clients do not share files, do not remove corrupted files, or quickly give up when the system is under attack. (ii) To launch a successful attack against a p2p network, it is insufficient to only transmit false information; the attackers must either invest in their own infrastructure or exploit software vulnerabilities in order to commandeer the resources of otherwise non-malicious peers. (iii) Structured p2p systems are more resilient than hierarchical p2p systems as the additional protocol functionality of nodes in the first-level of the hierarchy provides an acute DoS vulnerability. (iv) In both cases, system goodput degrades tremendously (hyperexponentially fast) with the number of malicious nodes, when users select to download files from the peer with best-advertised download time. (v) Reputation systems are largely ineffective, even with a very small number of false negatives. (vi) Randomization techniques are indeed able to transform the system's resilience from a devastating hyperexponential scaling to a more resilient linear scaling. Unfortunately, randomization severely hinders performance when no attackers are present.

8. REFERENCES

[1] L. Adamic, R. Lukose, A. Puniyani, and B. Huberman. Search in power-law networks. *Physical Review E*, 64:46135–1–7, 2001.

[2] R. Albert, H. Jeong, and A. Barabasi. Error and attack tolerance in complex networks. *Nature*, 406:378–382, 2000.

[3] BBC News. File swappers fight back. May 11, 2003, http://news.bbc.co.uk/1/hi/technology/3013065.stm.

[4] M. Castro, M. Costa, and A. Rowstron. Should we build gnutella on a structured overlay? In *HotNets*, 2003.

[5] M. Castro, P. Drushel, A. Ganesh, A. Rowstron, and D. Wallach. Secure routing for structured p2p overlay networks. In *OSDI*, 2002.

[6] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Comm. of the ACM*, 24(2):84–88, 1981.

[7] N. Christin, A. Weigend, and J. Chuang. Content availability, pollution and poisoning in peer-to-peer file sharing networks. In *ACM E-Commerce Conference*, 2005.

[8] I. Clarke. A distributed decentralised information storage and retrieval system. Master's thesis, Univ. of Edinburgh, 1999.

[9] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Design Issues in Anonymity and Unobservability*, 2000.

[10] R. Dingledine, N. Mathewson, and P. Syverson. Reputation in p2p anonymity systems. In *Economics of P2P Systems*, 2003.

[11] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *ACM SOSP*, 2003.

[12] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *NOSSDAV*, 2003.

[13] K. Hildrum, J. D. Kubatowicz, S. Rao, and B. Y. Zhao. Distributed Object Location in a Dynamic Network. In *ACM Symp. on Parallel Algorithms and Architectures*, 2002.

[14] A. IT. Music industry raids KaZaA offices. February 6, 2004. http://www.afterdown.com/news/archieve/4948.cfm.

[15] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *World Wide Web Conference*, 2003.

[16] J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in p2p file sharing systems. In *IEEE INFOCOM*, 2005.

[17] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the XOR metric. In *IPTPS*, 2002.

[18] T. Moreton and A. Twigg. Trading in trust, tokens, and stamps. In *Economics of P2P Systems*, 2003.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *ACM SIGCOMM*, 2001.

[20] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Middleware*, 2001.

[21] S. Chartrand. New way to combat online piracy. The New York Times, May 17, 2004.

[22] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy. An analysis of Internet content delivery systems. In *OSDI*, 2002.

[23] D. Schweitzer. *Securing the Network from Malicious Code: A Complete Guide to Defending Against Viruses, Worms, and Trojans*. John Wiley and Sons, 2002.