Characterizing Graph-Theoretic Properties of a Large-Scale DHT: Measurements vs. Simulations

Hani Salah, Stefanie Roos, and Thorsten Strufe

TU Darmstadt, Germany
{hsalah, roos, strufe}@cs.tu-darmstadt.de

***Abstract*—The widely used distributed hash table (DHT) in KAD is commonly analyzed and optimized based on partial measurements and simulation results, which are limited in scope and subject to simplification. An accurate characterization, however, is vital for a thorough understanding and effective enhancement.**

Analyzing and comparing complete real graphs collected from a large-scale measurement campaign as well as synthetic graphs generated by a novel simulation model, we study their degree distributions as well as resilience in face of random departure and targeted attacks. Our results show that the online KAD graph, although scale-free, is highly robust not only to random departure, but also to targeted attacks, making it suitable for distributed applications requiring a high resilience. Resilience to random departure and shape of degree distribution are well modelled by the simulations. However, due to a greatly increased ratio of stale routing information, the complete graph in the real system is much more vulnerable to targeted attacks compared to estimations based on simulative results.

***Index Terms*—Measurements, Simulation, Graph-theoretic Properties, KAD.**

I. Introduction

Structured peer-to-peer (P2P) systems in the form of distributed hash tables (DHTs) are widely used today. They have been integrated during the last years into multi-million user file-sharing applications like BitTorrent[1] and eMule[2], and they have been also employed recently as communication infrastructures for botnets [1]. Measuring and modelling these systems has been an attractive and challenging research problem during the last decade (e.g. [2]–[6]). Their accurate characterization is essential to understand them, to give guarantees on their performance, and to improve their quality.

This work focuses on the characterization of graph-theoretic properties of large-scale DHTs, via a case study. In particular, we have chosen the KAD network[3] as implemented in the eMule client for our exemplary analysis. KAD is not only one of the most popular DHTs, but it has also been extensively treated in research (e.g. [3], [7]). We consider three important properties: (*i*) the degree distribution, (*ii*) the graph resilience to *failures*, i.e. random removal or departure, and (*iii*) the graph resilience to *targeted attacks* that remove the highest degree nodes from the network.

These properties represent predominant concerns when designing distributed systems with outstanding reliability and availability features. The degree distribution is very important to understand the network structure, and it is closely related to routing durations and the graph resilience. The accurate analysis of nodes' *in-degrees* over the *entire* graph, for instance, can be utilized for the detection and mitigation of targeted eclipse attacks and other anomalies, more efficiently than prior approaches [8]. The resilience, on the other hand, provides direct information about the quality of the system graph for communications. For instance, the resilience is of interest to determine whether a certain DHT is suitable for botnet command and control or not [4], [9].

Though the properties of interest reveal important information, they have not been analyzed before on *real* DHT graphs. Their computation requires knowledge of the *entire* graph, which is highly challenging [2], [5] and has not been provided by earlier measurements. Furthermore, previously accepted results from simulations (e.g. [4], [9]) are limited by small network sizes, restricted node degrees, and simplified churn behaviour.

We summarize our contributions as follows: we characterize graph-theoretic properties of KAD through: (*i*) analyzing *both* measured snapshots of the *entire* KAD graph as well as simulated KAD graphs, (*ii*) *comparing* the properties of measured graphs to simulated graphs at both a *qualitative* and a *quantitative* level, and (*iii*) identifying the reasons for the observed disparities. We collect the real graph snapshots using our accurate KAD crawler [5]. We generate the simulated graphs using a novel simulation model which extends the very popular simulator for structured P2P systems, *OverSim* [10], by an accurate model of the KAD protocol exactly as implemented in eMule. The simulation model generates KAD graphs of different sizes, and it models node behaviour using the standard churn models of OverSim. Our resilience analysis is based on [11], [12]. However, we are the *first*, to the best of our knowledge, to transform their approach to unidirectional graphs, thereby making it applicable to KAD.

Our results show that both the measured graphs and the simulated graphs are highly resilient to failures and targeted attacks. However, while the simulations exhibit similar qualitative behaviour as the real-world system, there is a strong disparity on the quantitative level between the results obtained from the measurements and those obtained from the simu-

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

[2]http://www.emule-project.net/

[3]We use the terms *network* and *overlay* interchangeably in this paper.

lations. In particular, at most 30% of the total number of nodes are inactive in the simulations, whereas 88% of the entries seen in the routing tables during the measurements were stale. As a consequence, the average degree of all nodes is drastically reduced in the measured graphs. The resilience to targeted attacks is lower for the measured graphs, whereas the resilience to failures is closely modelled by the simulations. The shapes of the degree distributions are similar, though the simulations show fewer high-degree nodes and other irregularities. We attribute these disparities to: (*i*) the simplified churn models available in the simulator, (*ii*) the manipulated clients in the real system, and (*iii*) the restricted network sizes of the simulations.

The remainder of this paper is structured as follows. We discuss the related work in Sec. II. Then, Sec. III introduces KAD, our measurement methodology, and our simulation model. Next, Sec. IV presents the analysis results of graph-theoretic properties of the KAD network. Finally, Sec. V concludes the paper.

II. Related Work

During the last years, a considerable number of research work focused on P2P systems, e.g. see [3], [6], [13]–[16], and the references therein. However, very few studies considered global graph-theoretic properties. Existing studies are concerned with protocol-specific analysis, whereas our goal lies in analyzing global graph-theoretic properties in both reality and models. In the remainder of this Section, we only consider prior work studying graph-theoretic properties in large-scale P2P systems.

Early work on characterization of P2P overlays considered the unstructured P2P systems Gnutella [2], [17], [18] and Kaaza [19]. However, unstructured overlays are highly different from structured ones, which allow much more efficient content discovery and are hence used more commonly for that purpose. To the best of our knowledge, the only two papers that study global graph properties of structured P2P overlays are [4] and [9]. However, these papers are based on simulations only, and used small overlay sizes, whereas our study uses both real-world measurements and simulations.

The only comprehensive measurement study in a similar area is the characterization of routing tables in KAD by Stutzbach et al. [7]. Though their focus is the aliveness and completeness of routing tables, they indirectly sample the out-degree distribution. However, their analysis is based on small samples of routing tables, and hence their work cannot capture the in-degree, nor the resilience. Capturing these properties requires the knowledge of all routing tables.

III. Methodology

In this Section, we start with a brief description of KAD. We then give an overview of our measurement methodology and finally introduce our simulation model.

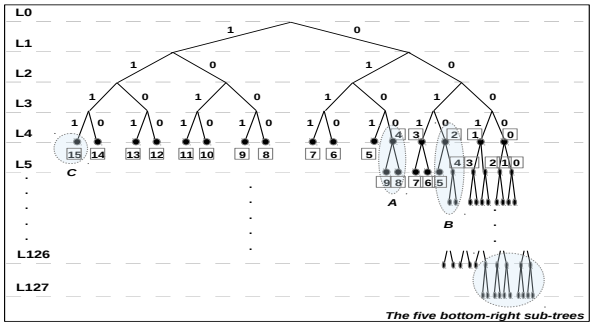


Fig. 1: The routing table structure of KAD (adapted from: [22]). It encodes the distances in the ID-space: The top-left part stores the most distant contacts, while the bottom-right part stores the closest ones. Each leaf (e.g. ***C***) represents a bucket of contacts. ***A*** and ***B*** are examples of two-bucket and three-bucket sub-trees, respectively.

A. Background on KAD

KAD is a Kademlia-like [20] structured P2P system used as a discovery overlay in the popular P2P file-sharing system eMule. It facilitates content delivery by enabling nodes to locate content in a distributed fashion requiring a low communication overhead. KAD has been also used as a communication infrastructure for botnets [4], [21].

Every node in KAD has a 128-bit identifier, called the *KAD ID (KID)*, representing a unique position in the *ID-space*. The KID of a node is randomly generated, persistent, and acquired by the node the first time it joins the overlay. The *distance* between two objects (nodes or files) is defined as the *XOR* of their identifiers.

The Routing Table*:* Every KAD node v stores information of other nodes (also called *contacts*) in a tree-like *routing table*. As illustrated in Fig. 1, the routing table has up to 128 levels (indexed 0-127). Starting from the fourth level, every tree level includes *buckets* of contacts. Every bucket can hold up to $k = 10$ contacts. Nodes are assigned to buckets based on the prefix of their distance to v. The five right-most buckets split further to store new contacts in case the original bucket is full. Correspondingly, buckets with the same common prefix and less than k contacts are merged.

The Routing Process*:* KAD's routing protocol applies iterative parallel look-ups realized by two UDP-based routing packets: *Kademlia_request* (KAD_REQ) and *Kademlia_response* (KAD_RES). The KAD_REQ includes the address of the destination node, the number of requested contacts (β), and the *target-ID* T. To find a route from a source node S to a target-ID T, S picks from its routing table and previously encountered peers α contacts that are closest to T. A KAD_REQ is sent to the α contacts in parallel. Every queried node, if available, answers by a KAD_RES including contact information of β possible next-hop contacts. This process continues *iteratively* till S discovers contacts very close to T or it reaches T itself.

B. Capturing Graph Snapshots of the KAD Overlay

Measurement Methodology*:* We use our crawler [5] to capture graph snapshots of the KAD overlay. The crawler performs two interleaving tasks: (*i*) discovering online nodes, and

(*ii*) downloading their routing tables. The evaluations [5] have shown the crawler's ability to capture graph snapshots with very high completeness degree in a very short time, allowing to achieve highly accurate and representative snapshots. Here we briefly describe the novel techniques applied in the crawler.

Similar to [23], we partition KAD's ID-space into equal-sized non-overlapping ID-sections and assign them *evenly* to crawling machines. In this way, every crawling machine works exclusively on its own assigned ID-section, and does not need to synchronize with other crawling machines. This design facilitates higher crawling rates and reduces traffic congestion and packet losses.

The crawling algorithm basically works as follows: Every time the crawler starts, it first downloads routing tables of a number of previously discovered nodes. In parallel, it extracts new contacts from downloaded routing tables and adds them to the list of known nodes. This procedure continues till the crawler discovers all contactable nodes and downloads their whole routing tables.

Our crawler outperforms prior KAD crawlers (e.g. *Blizzard* [3], *kFetch* [7], and [23]) in terms of the time and the number of crawling packets that are required to download routing tables. This achievement is mainly attributed to two novel techniques.

The first technique divides the routing tree into sub-trees of buckets. In Fig. 1, the sub-trees *A* and *B* are examples of two-bucket and three-bucket sub-trees, respectively. This division enables the crawler to download contacts of up to three buckets per KAD_REQ, instead of only one (as in prior crawlers). Note that prior crawlers assume that β (number of requested contacts in KAD_REQ) can be set only to 2, 4, or 11. However, β in eMule code can be between 0-31.

Our crawler uses the KAD *routing packets* rather than the *bootstrap packets* (like *Rememj* [24]) for two reasons: First, bootstrap responses return *random* contacts, which does not assure the coverage of the whole routing table (and thus does not assure the coverage of the whole overlay). Second, each bootstrap response includes only 20 contacts, which is fewer than the number of contacts that *can be* included in a single routing response.

Our second crawling technique attempts to avoid querying sub-trees of empty buckets. Towards this end, the crawler first queries the destination node's five lowest sub-trees (see Fig. 1). Next, it shifts to the top of the tree and starts downloading sub-tree buckets from top-left towards bottom-right. Note that the returned contacts get closer to the querying node's KID with each query. The crawler continues sending queries till the previously learned five *closest* contacts are encountered. At this point, the crawler stops sending queries because there are no closer contacts remain to discover.

C. Simulating the KAD Protocol

Existing popular P2P simulators (e.g. PeerSim [25], PeerfactSim.KOM [26], and OverSim [10]) implement different structured P2P systems, including Kademlia-like ones, but none of them implements KAD. Although KAD is derived from Kademlia, there are several differences between the two protocols [27] which make developing a new simulation model for KAD more straightforward than adapting an existing Kademlia model.

Towards this end, we developed a novel simulation model of KAD, and integrated it as a new P2P protocol in OverSim, a widely used OMNeT++ based simulator[4]. The model implements the protocol design and settings exactly as the eMule client software. Hence, our simulation model can be expected to represent the real system very accurately.

During the study, we performed simulations on three overlay sizes: 1000, 5000, and 10000 nodes, and experimented the three churn models that are available in OverSim: No Churn, Lifetime Churn (LC), and Pareto Churn (PC).[5] Due to space limitations, Sec. IV shows only results from simulations applying PC and overlays of size 10000. The results of simulations with LC are very similar and the impact of the overlay size is discussed. The results are averaged over ten simulation runs. Given that the results we achieved for simulated graphs applying same settings show very high similarity, we believe that ten runs are enough to conclude.

IV. Graph-Theoretic Properties

In this Section, we first introduce the properties of interest and our measured datasets. We then present our analysis for global graph-theoretic properties of the KAD graph.

A. Introduction of Graph Metrics

A graph $G = (V, E)$ consists of a set V of *nodes* (or *vertices*) and a set $E \subset V \times V$ of edges (or *links*). In the following, we denote nodes by u and v. Note that the KAD graph is *unidirectional*, meaning that $(u, v) \in E$ (i.e. a routing entry in u's routing table towards v) does not imply $(v, u) \in E$. In addition, we denote the expected value of a random variable X by $\mathbb{E}(X)$.

***Degree Distribution**:* The *in-degree* of a node v in a graph is $d_-(v) = |\{(u, v) \in E : u \in V\}|$, the number of incoming edges. Correspondingly, the *out-degree* $d_+(v) = |\{(v, u) \in E : u \in V\}|$ is the number of outgoing edges of v. The *degree* $d(v) = d_-(v)+d_+(v)$ is the sum of in- and out-degree. The *(in-/out-)degree distribution* D of a graph G maps each non-negative integer k to the fraction of nodes in G that have (in-/out-)degree k.

***Routing Table Completeness**:* In addition to the actual degree deg, we compute the expected out-degree assuming that a maximal number of nodes is contained in the routing table. More precisely, let $X_{n,p} \sim B(n-1, p)$ be a binomially distributed random variable with n trials and success probability p. The expected out-degree is the sum of entries in all routing table buckets. Let n denote the number of nodes, and p

[4]The simulation code is available at: https://www.p2p.tu-darmstadt.de/fileadmin/user_upload/Group_P2P/share/kad_simul.tar.gz

[5]For more information about the churn models implemented in OverSim, the reader is referred to: http://oversim.org/wiki/OverSimChurn.

the fraction of KIDs a bucket is responsible for. The expected number of entries $\mathbb{E}(C_{n,p})$ in the bucket is

$$\mathbb{E}(C_{n,p}) = \sum_{i=1}^{10} \left(1 - \sum_{j=0}^{i-1} \binom{n-1}{j} p^j (1-p)^{n-1-j} \right) .$$

Hence, $\mathbb{E}(D)$ is given by

$$\mathbb{E}(D) = 11\mathbb{E}\left(C_{n,0.5^4}\right) + 5\sum_{i=5}^{128} \mathbb{E}\left(C_{n,0.5^i}\right) .$$

We compute the ratio $r = \mathbb{E}(D)/deg$ in order to characterize the difference to maximally full routing tables. The measure is a good indicator for the impact of incomplete knowledge and outdated information, i.e. stale entries, on the completeness of the routing tables.

Resilience: We define the resilience of a graph, with regard to a removal strategy R, to be the fraction of nodes that need to be removed, so that no giant connected component remains. At this point, the graph breaks into small partitions, each containing less than half of the remaining nodes. In agreement with similar studies (e.g. [28]), we consider random removal as a model for *failures*, and removal of the nodes with the highest (in-/out-)degree as a model for *targeted attacks*.

Our approach to compute the resilience is based on the approach of [11], [12], which uses the degree distribution of the graph's nodes. This approach requires a computational overhead of $\mathcal{O}\left(\min\{m_+ m_-, n\}\right)$, where $m_{-/+} = \max_{v \in V} d_{-,+}$ is the maximum in- and out-degree. However, the approach in [11], [12] is designed for bidirectional graphs, and therefore we first need to transfer it to unidirectional graphs before adopting it in our analysis. Another approach to compute resilience is to simulate it by iteratively removing nodes from the graph and then recomputing the giant component. We did not use this approach since it is computationally expensive and thus does not scale to graphs as large as the measured KAD graphs.

For unidirectional graphs, the degree distribution is modelled as a random vector $D = (D_-, D_+)$. The key observation is that for a given D, G almost surely has a giant connected component if

$$\mathbb{E}(D_-, D_+) - \mathbb{E}(D_+) > 0 .$$

Furthermore, if the above quantity is strictly less than 0, G almost surely does not have a giant connected component [29], [30].

In order to obtain the resilience under a removal strategy R, the degree distributions $\tilde{D}^p = (\tilde{D}^p_-, \tilde{D}^p_+)$ after removing a fraction p of nodes are derived. Then, we determine the fraction p that needs to be removed such that no giant component exists as

$$\mathbb{E}(\tilde{D}^p_-, \tilde{D}^p_+) - \mathbb{E}(\tilde{D}^p_+) = 0 . \tag{1}$$

The derivation of $\tilde{D}^p_{-/+}$ and the solution of Eq. 1 are excluded due to space limitations. However, the individual steps for deriving resilience to failures and targeted attacks are analogous to the derivation in bidirectional graphs given in [11] and [12], respectively. In the following, we hence only give the formulas for computing the critical percentage p for failures and targeted attacks removing the highest degree nodes. For failures, Eq. 1 resolves to

$$(1-p)\mathbb{E}(D_-, D_+) - \mathbb{E}(D_+) = 0 . \tag{2}$$

For targeted attacks removing nodes of largest degree, let $r(p,k,j)$ be the fraction of removed nodes with in-degree k and out-degree j, and $\bar{F}_D(d)$ be the probability of a node to have a degree of at least d. For a fraction p of removed nodes with the highest degree, we have

$$r(p,k,j) = \begin{cases} P(D=(k,j)), & \bar{F}_D(k+j) \leq p \\ 0, & \bar{F}_D(k+j+1) \geq p \\ \frac{p - \bar{F}_D(k+j+1)}{\bar{F}_D(k+j+1) - \bar{F}_D(k+j)}, & otherwise. \end{cases}$$

The fractions p_- of removed outgoing edges is then given by $p_- = \sum_{k=1}^{m_-} \sum_{j=1}^{m_+} r(p,k,j) \frac{k}{E(D_-)}$, and analogously for p_+. With the above terminology, Eq. 1 is transformed to

$$\begin{aligned} &(1-p_-)(1-p_+) \\ &(\mathbb{E}(D_-, D_+) - \mathbb{E}(D_+) - 2kjr(p,k,j)) = 0, \end{aligned} \tag{3}$$

so that p can be computed numerically from Eq. 3 and the definition of p_- and p_+.

B. Measurement Datasets

We performed hundreds of partial crawls, and 65 refined full crawls over the whole KAD network, at different periods between April 2012 and September 2013, covering different times of the day. On average, we observed 452,740 concurrent active (i.e. online) nodes, and 3,812,030 distinct routing table entries (including stale ones).

Due to space limitations, the measurement results that we discuss in Sec. IV-C and Sec. IV-D are based on a single measured graph. However, the exemplary graph is representative to the whole dataset since the results that we obtained from other measured graphs are very similar. In particular, we consider a recent graph snapshot captured on 12 September 2013 at 20:00 GMT. The graph includes 498,765 active nodes and 4,194,667 distinct routing table entries. This means that about 88% of routing table entries in this graph are stale. The graph also contains 65,623 nodes that use multiple KIDs, mainly due to manipulation of the client software [6].

In our analysis, we extract two graphs from each measured or simulated graph: The first includes only *active* (i.e. online) nodes, and the second includes *all* nodes, i.e. both active nodes and stale routing table entries. For the measured graphs, we further extract two other graphs from each of the aforementioned graphs: One includes nodes that use multiple KIDs and the other excludes them. Due to their disparities, our analysis considers each of the four extracted graphs separately.

C. Degree Distribution Analysis

The degree distribution is very important to understand the network structure, and is closely related to routing durations and the graph resilience. In this Section, we first give an overview of the degree distributions of the measured graphs. In particular, we analyze the impact of stale entries and manipulated clients on the graph. Secondly, we compare the measured and the simulated graphs, leading to the conclusion that the stale entry rate is essential for the average degree, the completeness of the routing tables, and even the shape of the degree distributions.

For the degree distribution of the KAD graph, we have the following expectations: (*i*) The out-degree of a node is bound by the maximal number of routing table entries, hence we expect many nodes with a similar out-degree and none with a particular high out-degree. The in-degree, on the other hand, is unbound. Stable nodes attract more and more links, so that we expect a power-law degree distribution. (*ii*) The degree distributions of the active graph and the graph that includes all routing table entries are highly different, given the large fraction of stale routing table entries. The graph including all routing table entries is bound to have a lower average degree and more nodes with a low degree corresponding to stale entries, but also a higher maximal degree, because all entries in an active node's routing table are counted. (*iii*) The degrees of nodes that use multiple KIDs are potentially higher than the degrees of ordinary nodes, since they can be expected to have neighbors from a large fraction of the ID-space. However, we expect the influence of multi-KID nodes to be marginal because they only account for a small fraction of the graph.

Fig. 2 shows the degree distributions of the four graphs extracted from the exemplary measured graph. The results agree with our expectations: More precisely, the out-degree distribution (Fig. 2(a)) increases up to a mode, and drops rapidly afterwards when the capacity of the routing table is reached. Nodes with a low out-degree are the nodes that have not filled their routing tables yet. The in-degree distribution (Fig. 2(b)) follows a power-law, as expected. Consequently, the degree distribution (Fig. 2(c)) is asymptotically power-law, but also shows the mode of the out-degree distribution at roughly degree 1000.

As for the impact of stale entries and manipulated clients, we observe a drastic impact of stale entries and a low impact of manipulated clients. The out-degree distribution when considering only active nodes shows higher probabilities up to a routing table size of about 1000. However, the out-degree distribution considering all routing tables' entries shows a higher probability for large degrees. Note that when including all entries, there are 88% of nodes with out-degree 0 (i.e. inactive nodes). For the in-degree distribution, on the other hand, the graph containing all routing table entries shows a considerable higher fraction of nodes with a low in-degree, which are frequently stale entries. Nodes having multiple KIDs do not change the general shape of the curves, but lead to a higher degree, as expected. In particular, these are the nodes with the highest in-degree.

In the following, we compare the degree distributions and the average degrees of measured graphs to simulated graphs. We exclude nodes with multiple KIDs, because they do not appear in the simulations. Our expectations are as follows: (*i*) The simulated graphs should have degree distributions relatively similar in shape to the degree distributions of the measured graphs, although shifted due the lower number of nodes in the simulations. (*ii*) Since more than 70% of the nodes in the simulated graphs are active, in contrast to the 12% in the measured graphs, removing active nodes is bound to have a lower impact on the simulated graphs. Hence, the disparities of average out-degrees, the ratio r of the actual out-degree, and the expected out-degree, between the complete graph and the active graph are expected to be much lower in simulations than in measurements.

The out-degree distribution shows the characteristic drop at the maximal routing table size. However, note that there is a slight difference in shape with regard to the in-degree distribution: Whereas the in-degree distribution of the measured graphs exhibit a constant negative slope, the simulated graphs result in local extrema. The local extrema persist when considering the active graphs[6].

Table I displays the average degrees and the ratio r for simulated and measured graphs. The results confirm our expectations: The impact of the high stale entry rate in measured graphs on the average degree is drastic. Thus, the average out-degree in the measured graphs of roughly 4 million nodes is less than 60, which is below the average out-degree of 90, observed in a simulated graph of only 1000 nodes.

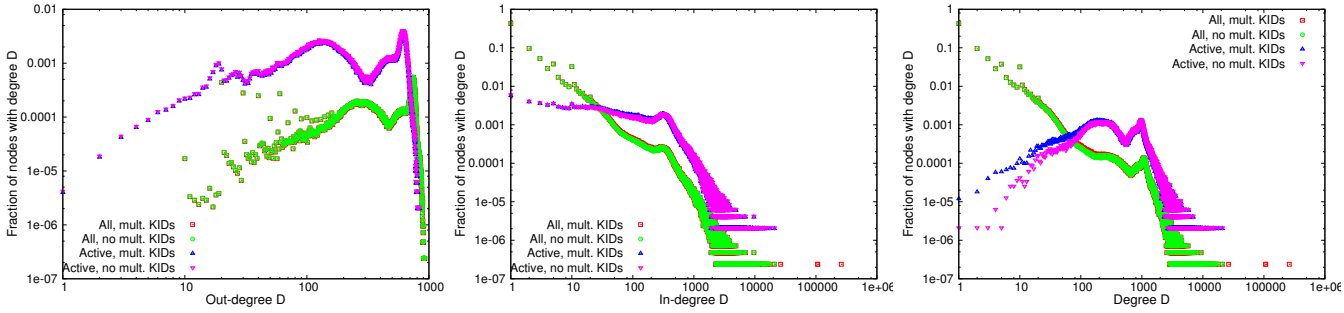
When only considering active nodes, the average degree is higher for measurements, as expected, given the larger size of the networks. Similar results hold for the disparity r between perfect and actual routing tables. For simulated graphs, r is between 3 and 3.5, about a fifth of the corresponding value for the measured graphs when considering all entries. However, for the active graphs, the ratio r between simulated graphs and the measured graphs is comparable, even slightly lower for the measured graphs.

In summary, measured graphs exhibit a large number of inactive nodes, in comparison with simulated graphs. Stale entries create various graph-theoretic properties that are not well modelled by the simulations.

D. Resilience Analysis

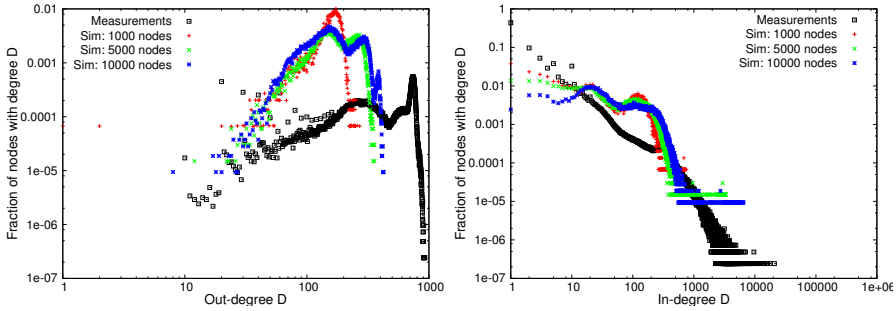
In Sec. IV-C, we have analyzed the degree distribution which has an impact on the system performance, but it does not directly provide information about the quality of the system graph for communications. These information can be obtained through analyzing the graph resilience, a predominant concern when designing any distributed system, especially for highly dynamic systems. In this Section, we analyze the resilience of KAD graphs to failures, i.e. random removal or

[6]Graphical results of degree distributions considering only active nodes are excluded due to space limitations. However, the *shapes* of degree distributions are similar to those of complete graphs (Fig. 3).



(a) Out-Degree (b) In-Degree (c) Degree

Fig. 2: Degree distribution of an exemplary measured KAD graph
(**All:** both active nodes and stale routing table entries are included; **Active:** only active nodes are included; **Mult. KIDs:** nodes with multiple KIDs are included)



(a) Out-Degree (b) In-Degree

Fig. 3: Degree distributions of an exemplary measured KAD graph and three different-sized simulated KAD graphs
(All routing table entries are included)

departures, as well as to targeted attacks, and compare the results of measured and simulated graphs to each others. The computations are executed according to Eq. 2 and Eq. 3.

In KAD, the resilience to failures is expected to be high and increasing with the graph size, as shown for scale-free graphs in [11]. As a consequence, we expect measured graphs to be more resilient than simulated ones. The difference between the active graph and the graph including all nodes is likely to be small, given that both are scale-free and the result holds for all scale-free graphs. The resilience of scale-free graphs to targeted attacks, on the other hand, has been shown to be low [12]. However, their theoretical analysis is only asymptotic, and our simulated graphs have a very high average degree in comparison with the graphs in [12]. For this reason, we expect that a high fraction of nodes needs to be removed in order to destroy the giant component. Of course, the fraction is much lower than when applying random removals. The active graph should be more resilient to targeted attacks, because the disparity between nodes with a low degree and a high degree is less pronounced. More precisely, the hubs are less influential because they have less connections. Simulated graphs are expected to be more resilient to attacks. Their maximal degree is lower and hence the impact of the removed hubs is much lower. As for the manipulated clients in the measured graphs, they tend to show a higher degree and are hence bound to reduce the resilience to targeted attacks.

Table II shows the resilience of both measured and simulated graphs, on both active nodes and all nodes. The provided values characterize the fraction of nodes that need to be removed until there is no giant connected component. The results confirm our expectations: The resilience to failures is well above 99% in all considered graphs. In addition, the resilience slightly improves with the graph size, being at its maximum for the measured graphs. The difference between the active graph and the graph including all routing table entries is negligible in the case of failures. For targeted attacks, however, the active graph is much more resilient, due to the lower impact of hubs, as discussed above.

Deriving the influence of the number of nodes is complex: When considering all nodes, an increase of the graph size correlates with an increased resilience for the simulation. Here, the increased average degree seems to be the dominating factor. Naturally, the resilience to targeted attacks seems very low in case of the measured graphs. However, given that there are 88% of stale entries, needing to remove 12% of the nodes means that the majority of active nodes has to be removed. So, when considering only active nodes the resilience is in general higher, but deceases with the graph size. The average degree is already high, so that the stronger proportional increase of maximal degree is dominating. As a consequence, high-degree nodes become more important when the graph is larger, i.e. their removal destroys the structure faster.

Including nodes with multiple KIDs reduces the resilience of the measured graphs to targeted removals by about 2%, as expected. Hence, the resilience of a graph is influenced by anomalies such as manipulated clients, which cannot be

TABLE I: Average out-degree vs. expected out-degree in measured and simulated graphs

	All			Active		
Graph	d	$E(D)$	r	d	$E(D)$	r
PC-1000	87.02	285.7	3.28	129.45	271.11	2.09
PC-5000	112.00	401.86	3.59	155.13	382.85	2.47
PC-10000	141.00	451.87	3.2	162.81	431.04	2.65
Measured	56.3	887.45	15.76	363.76	733.86	2.02

TABLE II: Resilience values in measured and simulated graphs

	All		Active	
Graph	Random	Targeted	Random	Targeted
PC-1000	0.9928	0.5428	0.9929	0.9194
PC-5000	0.9944	0.5776	0.9944	0.9119
PC-10000	0.9951	0.7359	0.9951	0.8860
Meas. (w/o mod. cl.)	0.9984	0.1100	0.9980	0.8625
Meas. (w. mod. cl.)	0.9984	0.1105	0.9980	0.8443

modelled accurately. In general, we see that predicting the resilience of large graphs by analyzing the protocol on a small scale does only provide qualitative information. It can be predicted that the resilience increases or decreases with the network size. However, the actual values are highly different in real-world networks. The results above show that the tendencies observed in simulated graphs are indicators for the resilience of measured graphs. However, the resilience of measured graphs can be much lower for targeted attacks. Anomalies such as nodes with multiple KIDs harm the resilience of the graphs to attacks. These findings confirm that graph-theoretic properties can be used to automatically detect anomalies and attacks.

Our results regarding the graph properties of measured KAD graphs are of interest on their own, regardless of their relation to simulated graphs. They indicate that the KAD graph structure is highly resilient to both random failures and targeted attacks. Hence, KAD is a suitable design for distributed applications requiring a high resilience such as critical infrastructures or botnets.

V. Conclusion

We performed an exemplary study on the characterization of global graph-theoretic properties in popular DHTs. Our study is performed on the KAD network, and the analyses show that KAD is highly resilient to both random and targeted removals, and hence it is suitable for distributed applications requiring a high resilience such as critical infrastructures or botnets. We found that simulations are able to capture tendencies and qualitative metrics. However, there were disparities between the measured and simulated graphs, particularly for the graph resilience to targeted attacks. The disparities can be explained by simplified churn models used in simulations, manipulation of the client software, and the difference in the network size.

Acknowledgements

This work was supported by the German Academic Exchange Service (DAAD) and Center for Advanced Security Research Darmstadt (CASED). We would like to thank Sven Frese and Oleksii Donets for their help with the simulations.

References

[1] G. Memon, J. Li, and R. Rejaie, "Tsunami: A parasitic, indestructible botnet on kad," *Springer P2P Networking and Applications*, 2013.
[2] D. Stutzbach, R. Rejaie, and S. Sen, "Characterizing unstructured overlay topologies in modern p2p file-sharing systems," *IEEE/ACM Transactions on Networking*, 2008.
[3] M. Steiner, T. En-Najjary, and E. W. Biersack, "Long term study of peer behavior in the kad dht," *IEEE/ACM Transactions on Networking*, 2009.
[4] D. T. Ha *et al.*, "On the effectiveness of structural detection and defense against p2p-based botnets," in *Proceedings of IEEE/IFIP DSN*, 2009.
[5] H. Salah and T. Strufe, "Capturing connectivity graphs of a large-scale p2p overlay network," in *Proceedings of HotPOST*, 2013.
[6] J. Yu *et al.*, "Id repetition in structured p2p networks," *The Computer Journal*, 2011.
[7] D. Stutzbach and R. Rejaie, "Improving lookup performance over a widely-deployed dht," in *Proceedings of INFOCOM*, 2006.
[8] T. Cholez *et al.*, "Detection and mitigation of localized attacks in a widely deployed p2p network," *Springer P2P Networking and Applications*, 2012.
[9] C. Davis *et al.*, "Structured peer-to-peer overlay networks: Ideal botnets command and control infrastructures?" in *Springer Computer Security-ESORICS*, 2008.
[10] I. Baumgart, B. Heep, and S. Krause, "Oversim: A flexible overlay network simulation framework," in *IEEE Global Internet Symposium*, 2007.
[11] R. Cohen *et al.*, "Resilience of the internet to random breakdowns," *Physical review letters*, 2000.
[12] ——, "Breakdown of the internet under intentional attack," *Physical Review Letters*, 2001.
[13] D. Carra *et al.*, "On the impact of incentives in emule: Analysis and measurements of a popular file-sharing application," *IEEE JSAC*, 2013.
[14] D. Stutzbach and R. Rejaie, "Characterization of p2p systems," *Handbook of Peer-to-Peer Networking*, 2009.
[15] ——, "Understanding churn in peer-to-peer networks," in *Proceedings of IMC*, 2006.
[16] S. Roos, H. Salah, and T. Strufe, "Comprehending kademlia routing-a theoretical framework for the hop count distribution," *arXiv preprint arXiv:1307.7000*, 2013.
[17] R. Matei, A. Iamnitchi, and I. Foster, "Mapping the gnutella network," *IEEE Internet Computing*, 2002.
[18] M. Jovanović, F. Annexstein, and K. Berman, "Modeling peer-to-peer network topologies through small-world models and power laws," in *Proceedings of TELFOR*, 2001.
[19] J. Liang, R. Kumar, and K. W. Ross, "The kazaa overlay: A measurement study," in *Proceedings of IEEE annual computer communications workshop*, 2004.
[20] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *Proceedings of IPTPS*, 2002.
[21] G. Memon, J. Li, and R. Rejaie, "Tsunami: A parasitic, indestructible botnet on kad," *P2P Networking and Applications*, 2013.
[22] P. Wang *et al.*, "Attacking the kad network," in *Proceedings of SecureComm*, 2008.
[23] Q. Wu and X. Chen, "Advanced distributed crawling system for kad network," *Journal of Computational Information Systems*, 2011.
[24] J. Yu and Z. Li, "Active measurement of routing table in kad," in *Proceedings of CCNC*, 2009.
[25] A. Montresor and M. Jelasity, "Peersim: A scalable P2P simulator," in *IEEE P2P*, 2009.
[26] D. Stingl *et al.*, "Peerfactsim.kom: A simulation framework for peer-to-peer systems," in *HPCS*, 2011.
[27] D. Mysicka, "Reverse engineering of emule," *Semester thesis, Swiss Federal Institute of Technology (ETH)*, 2006.
[28] C. Davis *et al.*, "Structured peer-to-peer overlay networks: Ideal botnets command and control infrastructures?" in *Computer Security-ESORICS*, 2008.
[29] M. Molloy and B. Reed, "A critical point for random graphs with a given degree sequence," *Random structures & algorithms*, 1995.
[30] M. Newman *et al.*, "Random graphs with arbitrary degree distributions and their applications," *Physical Review E*, 2001.