Efficient DHT attack mitigation through peers' ID distribution

Thibault Cholez, Isabelle Chrisment and Olivier Festor
MADYNES - INRIA Nancy-Grand Est, France
{thibault.cholez, isabelle.chrisment, olivier.festor}@loria.fr

***Abstract*—We present a new solution to protect the widely deployed KAD DHT against localized attacks which can take control over DHT entries. We show through measurements that the IDs distribution of the best peers found after a lookup process follows a geometric distribution. We then use this result to detect DHT attacks by comparing real peers' ID distributions to the theoretical one thanks to the Kullback-Leibler divergence. When an attack is detected, we propose countermeasures that progressively remove suspicious peers from the list of possible contacts to provide a safe DHT access. Evaluations show that our method detects the most efficient attacks with a very small false-negative rate, while countermeasures successfully filter almost all malicious peers involved in an attack. Moreover, our solution completely fits the current design of the KAD network and introduces no network overhead.**

***Index Terms*—DHT; KAD; Sybil attack; attack detection; attack mitigation; IDs distribution**

I. Introduction

In our previous work on the KAD DHT [2], we highlighted two vulnerabilities that allow attackers to take control over DHT entries through distributed and localized attacks. Firstly, the possibility for peers to freely choose their KADID allows them to be placed as close as they want to any given indexed content. Secondly, to provide the best possible performances when searching for the stored contents, the routing algorithm of KAD always tries to store a published content on the closest nodes found around the related DHT entry. Combining these two properties enables few distributed peers, which choose their position in the DHT to be extremely close to a given entry, to take control over the indexation of this content by attracting all the related requests.

Such a control enables several functions which raise both privacy and security issues for users:

- Monitoring all requests emitted for a given indexed content (keyword or file) to know who is sharing it or looking for it;
- Polluting a keyword with fake files or a file with fake sources to disturb the network or promote honeypots;
- Eclipsing DHT entries to remove the related contents from the P2P network.

Our goal is to design a defense scheme which can protect a real DHT against distributed and localized attacks. Two solutions are possible: avoiding malicious peers to freely choose their place on the DHT or detecting them before sending service requests. The first solution leads to the problematic of identity management in P2P networks against Sybil attacks. We show in Section II-C that existing solutions which strongly constrain peers' ID are not yet suitable for real applications. Moreover, changing the identity management in KAD would need deep modifications in its design and source code, breaking backward compatibility.

We present here a very efficient and innovative solution to mitigate DHT attacks. It consists in detecting DHT attacks by comparing the abnormal peers' ID distribution introduced around the targeted entry to the theoretical IDs distribution. Thanks to our solution, the fact to freely choose peers' ID is not a major weakness anymore but a way to detect abnormal behaviors. We also introduce a countermeasure preventing malicious peers to receive service requests for the target. Besides being able to detect and mitigate attacks, our solution involves no network overhead and fits all the constraints of the current KAD P2P network.

This paper is structured as follows. Section II presents the research works related to DHT security issues and particularly in KAD. Section III analyzes measurements of safe peers' ID distributions from the real KAD network and proposes preventive rules against attacks. Section IV describes and evaluates our solution to detect localized DHT attacks and the countermeasure to filter malicious nodes. Finally, Section V concludes the paper and outlines our future works.

II. Related work

A. The KAD DHT

KAD is a widely deployed (∼3 million concurrent on-line users) P2P network based on the Kademlia distributed hash table routing protocol [8]. It is implemented by the eMule and aMule clients, which allow users to share files on this network. Several research papers have been written about large scale monitoring of the KAD network, with either crawler based [13] or passive [9] approaches.

Each KAD node has a 128 bits "KADID" determining its position in the DHT. All routing tasks are based on the XOR metric used to evaluate the distance between two peers or between a peer and a content. The routing table is composed of groups of contacts in a binary tree so that the closer an ID is to the current node, the better its knowledge of this part of the DHT is. The *Search* process is used to achieve any service on the DHT and is composed of two steps. The first step aims at finding the best possible peers near a DHT entry. This routing is done in an iterative way with parallel

978-1-4244-6534-7/10/$26.00 ©2010 IEEE

lookups by using *KADEMLIA_REQ* requests to discover new peers closer to the targeted address at each iteration.

When the list of the closest peers is set, the second step of the *Search* process sends KAD service requests to the 10 best peers found. As a file sharing application, the purpose of the KAD DHT is to index files and keywords. When a new file is shared, the raw data and all the keywords composing its name are hashed separately with a MD5 function generating a KADID for each piece of information. Those KADIDs are then published in the DHT. The peers being able to index a file or a keyword are those that are close enough to the published hash. This distance is called the "tolerance zone" of a KADID, and is set to the first common 8 bits (the most significant). However, although the theoretical tolerance zone is big (a 8 bits common prefix enables 16K nodes for a 4M peers network), the nodes returned by the DHT routing process are always much closer to the target (usually $>$ 16 bits). The double indexing scheme allows a peer to retrieve sources for a file, given a set of keywords. To publish a file, two types of service requests are sent:

- the *KADEMLIA2_PUBLISH_KEY_REQ* requests: sent towards the hash of the keyword to link it to a file
- the *KADEMLIA2_PUBLISH_SOURCE_REQ* requests: sent towards the hash of the file to link it to a source (a peer sharing the file)

After accepting a publication request for a given resource, a peer is in charge of indexing this specific content, and answering the related search requests.

B. DHT security issues

The main well-known security issue of DHTs is the Sybil attack. As described by Douceur [5], it consists in creating a large number of fake peers called the "Sybils", and placing them in a strategic way in the DHT to take control over a part of it. Steiner et al successfully launched this attack on KAD [14]. They injected 2^{16} Sybils from a single computer in a small zone of the DHT, so that they were able to catch most of the publish and search requests for contents indexed within this zone. They also achieved more localized eclipse attacks making some keywords indexed on the Sybils disappear from the DHT. Another security threat of KAD consisted to overwrite the IP address of legitimate contacts in peers' routing table [15]. The KADID spoofing exploited at a large scale allowed attackers to partition the DHT and to do massive denials of service with few resources.

However, these two attacks were mitigated in the latest versions of KAD clients [1]. Several protection mechanisms have been introduced: 1) a protection against flooding reduces the efficiency of crawlers, 2) it is no longer possible to infect a peer's routing table with Sybils having the same or very close IP addresses (same /24 subnetwork), and 3) a three-way handshake prevents a peer to overwrite the IP address of another one in routing tables by claiming the same KADID. Classical Sybil attacks [14] on KAD which directly inject Sybils in routing tables became inefficient.

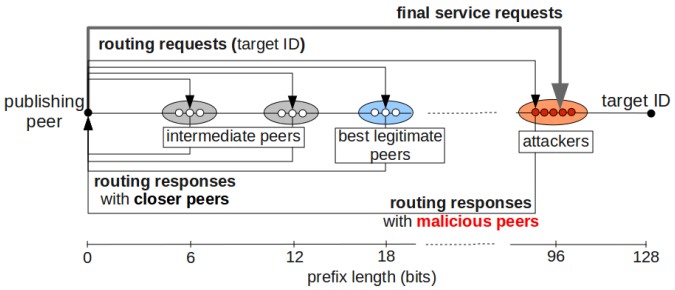

Fig. 1: Message exchange in a *Search* process under attack

Type	KADID	prefix
target	477221265829086C74988C40EFE63DAF	-
attacker	477221265829086C74988C4070D6E0F1	96 bits
normal	477229E3D7CFC729F337ABBB69C983C6	20 bits

TABLE I: Example of peers' ID

While protective rules have been set to protect the routing table from Sybil attacks, the lookup process remains unprotected, even in the latest versions. In fact, to control or disturb the indexation mechanism, [2] and [6] adopted a strategy focusing on the *Search* process instead of the peers' routing table. Few distributed peers (with different IP addresses) are placed closer than any legitimate peer to the targeted DHT entry as shown in Figure 1 and Table I. Additionally, to ensure that the inserted peers are found and attract all service requests, they cooperate to promote each other during the lookup process. In [2], the proposed architecture, HAMACK, allows us to monitor, eclipse or pollute any content stored in KAD with few resources ($\sim$ 20 nodes) while [6] proposed an efficient strategy to conduct eclipse attacks.

C. Proposed solutions

Many defense strategies described in the literature try to limit the Sybil attack. Proposed solutions are based on a strong identification of peers [4] or on a trusted central authority. In [11], the authors propose to bound the degree of overlay nodes by anonymous auditing to limit localized attacks. However, they also rely on a central certification to limit the number of Sybils. A strong identification being not adapted to many P2P applications, other strategies are possible to limit the Sybil attack. Some solutions consider that a distributed assignment with free identifiers is possible but, in compensation, must be verified by resource consuming proofs [10] or an underlying social network [16]. The most successfully completed protection against the Sybil attack [7] uses distributed certification coupled with a social network, but no studies so far apply the protection in a real network. Without limiting the number of Sybils, [3] improved the DHT lookup process under denial of service attacks performed through massive Sybil insertions. However, the authors do not consider the issue of localized attacks on DHT entries while their solution needs significant protocol modifications.

Even if defenses against Sybil attacks have been largely investigated, the proposed solutions are only partially convenient. Each of them has drawbacks: strong constraints to be defined at the start of the DHT (social network), incompatibility with an open network (central authority), or too much overhead (crypto puzzles). No solution meets the requirements of KAD which can not rely on a managed support infrastructure and needs a full backward compatibility with older clients. Those constraints limit the possible modifications on the DHT and make the design of a defense scheme very challenging.

III. Analysis of peers' ID distributions for safe DHT entries

Localized attacks are very efficient on the KAD DHT because peers found during the lookup process are always sorted by distance to the target, and then requested in that order with service requests. If 10 or more peers of an attacker are found to be the closest to a DHT entry, they will attract all the requests thanks to their proximity to the target. However, managing to place several peers very close to a DHT entry has clearly visible consequences. We demonstrate that a localized attack introduces both proximity and density abnormalities in IDs distribution around its target. Therefore, to detect DHT attacks, we consider the unusual peers' ID distribution resulting from the placement of these malicious peers. The knowledge of common ID distributions and the observation of a real distribution returned by a DHT lookup allow us detect if a DHT entry is under attack, without any network overhead.

A. Theoretical IDs distribution

In KAD, like in most DHTs, the normal behavior of legitimate peers is to randomly choose their ID (128 bits KADID). If this random distribution is uniformly random, the average distance between two IDs is only defined by the number of peers in the DHT. Moreover, we can compute the number of potential peers sharing a given prefix with a target ID and infer a probability for any peer found after a lookup process to legitimately exist.

Let F be the function giving the mean number of peers sharing at least x bits with a target ID and N be the number of peers in the network. In our practical application for KAD, we consider a realistic number of 4 millions peers taking part in the DHT at the same time.

$$F(x) = \frac{N}{2^x} \tag{1}$$

with $N = 4 \times 10^6$ and $x \in [1; 128]$. More precisely, the theoretical prefixes (number of common bits) distribution of peers' ID around a target ID follows a geometric distribution with parameter $p = 1/2$:

$$P(X = k) = (1 - p)^{k-1} p \tag{2}$$

However, even if IDs distribution is well known at the network scale, we do not know if the best contacts found during a lookup process on the DHT reflect this distribution. As a peer only has a partial view of the network through its routing table, it relies on a lookup (in KAD called *Search*) procedure to find the best peers on which publishing or searching contents. It is difficult to infer if the peers found after a DHT lookup are close to the best potential peers existing in the DHT (given by the random ID generation). Moreover, several parameters can affect the routing efficiency of a peer (uptime, distance to the target ID...) and need to be considered.

B. Real IDs distributions

To confront the results of the KAD lookup algorithm to the theoretical random ID distribution, we ran an experiment that consists in requesting many lookups on safe DHT entries and writing for each lookup the best contacts found. We measured these real peers' ID distributions at the end of the lookup process (just before the second step when specific service requests are sent to the 10 best contacts found). At this stage, many contacts have been found in the tolerance zone of the target but we focus our attention on the ID distribution of the 10 best peers only.

We published 100 files over 60 hours. To be sure that no attack is currently targeting our files, the two keywords composing each file name and the raw data are randomly generated. The default KAD behavior is to publish the keywords every 24 hours and the files every 5 hours resulting in approximately 1800 ($12 \times 100 + 3 \times 200$) publications on the DHT. For each publication, we monitored the contacts found by our client and computed the prefix shared between these contacts and the hash of the content to be published.

Figure 2 shows that the average common prefix size of the 10 best contacts found (which receive the service requests) is between 18 bits and 19 bits which is a very good result considering the number of potential peers sharing this prefix length in a 4 million peers DHT (between 15.25 and 7.6 from Table II). This result confirms the KAD routing efficiency since the 10 best peers found after the lookup process are nearly the 10 best peers actually existing in the DHT. Figure 2 also illustrates that the time spent in the network has no significant impact on the average common prefix of the 10 best contacts, despite the fact that the routing table consistency is improved over time. We can see a very small periodic variation of the average common prefix size. This small amplitude can be explained because KAD is mainly used in European countries and in China, the activity hours of these two regions being complementary.

We investigated several parameters potentially affecting the quality of the best found contacts. The DHT distance between the requesting peer and the target does not impact the routing efficiency. As seen in Figure 3, the average common prefix size of the best contacts remains stable. The lookup process probably hides the advantage that could be provided by the initial routing table when peers and contents are close in the DHT. In the same way, the type of published information (keyword or file), or the type of requested services (publish or search) have no impact on the quality of the found contacts.

Finally, the most important result concerning this experiment is given in Figure 4. For each publication, we counted

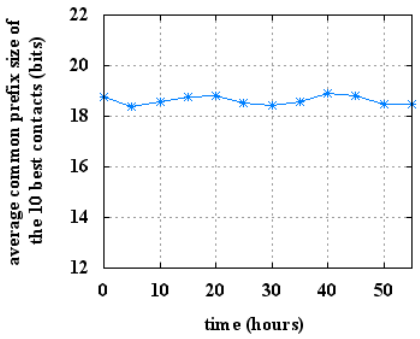

Fig. 2: Average common prefix of the 10 best contacts in time

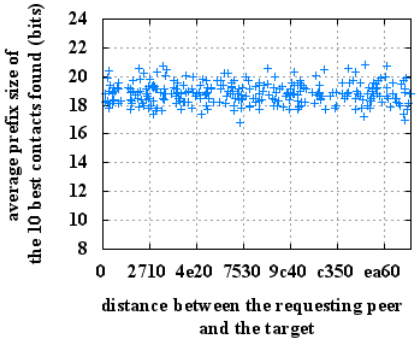

Fig. 3: Average common prefix of the 10 best contacts regarding the distance requester-target

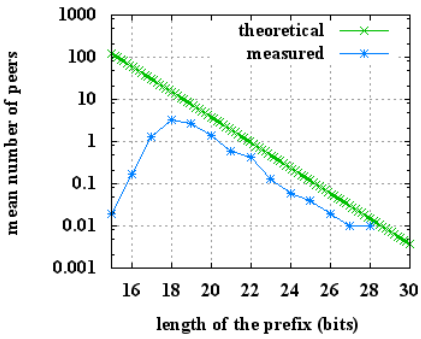

Fig. 4: Prefix distribution of the 10 best contacts

Common prefix length	Mean number of peer found	Theoretical number of peer existing (on 4 million)
15	0.0196	122
16	0.1667	61
17	1.2647	30.5
18	3.2255	15.26
19	2.6274	7.63
20	1.4118	3.81
21	0.6078	1.9
22	0.4118	0.95
23	0.1274	0.48
24	0.0588	0.24
25	0.0392	0.12
26	0.0196	0.06
27	0.0098	0.03
28	0.0098	0.015

TABLE II: Mean number of peers among the 10 best contacts found sharing a given common prefix size with a target

the number of peers among the 10 best peers sharing a given common prefix with the target ID. It appears that, after the 17 bits prefix, the distribution exactly follows the theoretical peers' ID distribution described in the previous subsection. So, the KAD *Search* procedure is efficient enough to give a representative view of the closest peers possible. Above all, this result shows that the theoretical random ID distribution is sufficient to characterize the results obtained in a real lookup process. The normal behavior does not need to be described by a learning-based approach. We will rely on this strong result to detect DHT attacks.

With the knowledge of the common peers' ID distribution, we first define some preventive rules to avoid the most obvious attacks. In a second phase, we present a metric that can accurately detect the remaining attacks.

C. Preventive rules

Recent Sybil attack protection mechanisms introduced in KAD [1] demonstrated that few passive rules can mitigate the simplest attacks, without significant drawbacks. With the same idea, the following protective rules applied to the lookup process make an attack more difficult to realize.

1) IP based protection: Like the protection against Sybil attacks, we apply the same protection rules, already set to protect the routing table, to the peers found during the lookup process. Basically, for a given publication on the DHT, we forbid several hosts from the same subnetwork to be responsible for this entry. When publishing a content on the DHT, a peer must check that all the emitted replicated requests for this content will be sent to peers from different subnetworks. In this way, an architecture that would be able to catch all the requests is harder to set up. This protection aims to distribute peers in charge of a DHT entry on the IP-network scale.

2) Discarding too close nodes: A second protection consists in setting a minimum distance between the target ID and the closest responsible peers, according to the normal peer behavior. As shown previously, we can assume that legitimate peers randomly choose their ID. So, we can define a threshold beyond which closer peers become very unlikely and suspicious to attack the DHT entry. Without any protection, any peer can index a content when sharing at least 8 bits with the target ID. In other words, any peer sharing between 8 bits and 128 bits with a target ID can be selected to receive final service requests. However, peers sharing a very high common prefix (i.e. between 28 and 128 bits) are unlikely as illustrated in Figure 4. So, we define a minimum distance between the target ID and the closest responsible peers of 28 bits, which modifies the range of the tolerance zone from [8;128] to [8;28]. The lookup process will discard any peer closer than 28 bits to the target without decreasing the DHT efficiency. Nevertheless, this protection is not sufficient as localized attacks involving many peers sharing 28 bits with the target can still manage to get all the requests.

IV. Detection and protection against localized attacks

A. Detecting attacks with Kullback-Leibler divergence

To detect attacks, we compare the 10 best peers distribution to the theoretical distribution previously described. The 10 best peers are the most significant to consider: as they are selected to receive service requests, they are the root of an attack.

The major difficulty is that the few best peers constitute a too small sample size to get results from the common statistic tools comparing distributions. We first tried the chi-square and the Kolmogorov-Smirnov methods to compare IDs distributions but they are unable to give interesting results in our situation. In our case, we noticed that a metric based on the Kullback-Leibler divergence (also called G-test) gives much better results to detect an attack. In fact, it is known to be more efficient for empirical distributions with a small number of observations [12]. Given the probability distributions M and T of a discrete random variable, the K-L divergence between M and T is defined as:

$$D_{KL}(M \mid T) = \sum_i M(i) log \frac{M(i)}{T(i)} \quad (3)$$

We define a discrete random variable taking 11 values representing the distribution of common prefixes between 18 bits and 28 bits. T represents the theoretical geometric distribution of the IDs, with parameter $1/2$. M is the prefixes distribution of the 10 best contacts found at the end of a lookup process. Table III shows the values defined for T and two distribution examples for M. To detect attacks, we do not consider IDs with a common prefix under 18 bits for two reasons. First, we showed in Table II that focusing on the 17 bits prefix or less would only permit an attacker to capture in average 1.26 out of the 10 requests, while being in competition with many legitimate peers. Second, as seen in Figure 4, the average distribution currently observed in KAD of the best contacts follows the geometric distribution $(1/2)$ from the 18 bits prefix. This bound can evolve to higher prefixes if the number of peers in the DHT increases (i.e. 19 bits if the number of peers doubles) or to lower if it decreases. Concerning the other bound of 28 bits prefixes, we mentioned previously why closer contacts should be filtered preventively.

The K-L divergence is a non-symmetric measure $D_{KL}(M \mid T) \neq D_{KL}(T \mid M)$. In our case, comparing the divergence of T from M is more significant to detect attacks. For each common prefix between 18 and 28 bits for which peers are found ($M(i) \neq 0$), the measured distribution for this prefix is compared to the theoretical distribution and added to the global K-L divergence. Highest prefixes (with small T values) and contacts grouped on a single prefix (resulting in a high M value) generate a big increase of the divergence which is relevant to detect attacks. The K-L divergence gives the advantage to be effective with a small sampling and its incremental computation can tell which value of the distribution increases the most the divergence, what will be useful to apply precise countermeasures in case of attacks. However, the K-L divergence does not give a probability of similarity between the two distributions but a relative value. So, we have to define which value of K-L divergence indicates an attack.

B. Evaluation with simulated attacks distributions

To define a detection threshold, we computed the K-L divergence for two sets of distributions representing two situations: a lookup process for safe or attacked DHT entries. The first set of distributions is based on the results of many lookups for randomly generated data, as described in Section III. The second set of distributions is simulated to represent attacks.

# of malicious peers inserted	# of prefixes targeted	Repartition of the peers	# of generated distributions
5	1	5	11
5	3	2-2-1	8
5	5	1-1-1-1-1	6
10	1	10	11
10	2	7-3	9
10	2	5-5	9
10	3	5-3-2	8
10	4	4-3-2-1	7
10	5	4-2-2-1-1	6
10	6	2-2-2-2-1-1	5
10	7	2-2-2-1-1-1-1	4
10	10	1-1-1-1-1-...-1	2

TABLE IV: Possible repartition of peers with common prefixes between 18 & 28 bits

1) Simulation of attacks: An attack on a DHT entry can lead to different peers'ID distributions, regarding the parameters (number of malicious peers involved, proximity to the target...) used by the attacker. Basically, an attacker must answer two questions: How many peers to insert? On which prefixes? As explained, our detection mechanism consider prefixes distribution of the 10 best peers sharing a common prefix between 18 bits and 28 bits. Because malicious peers can have a very high uptime, we consider that all inserted malicious peers are found during the lookup process. So, to simulate attacks distributions, we first initialize a distribution with, for each observed prefix size, a number of peers close to the real mean number provided by our observations. We then modify this set of peers by adding malicious peers according to different placement strategies (Table IV) and we compute the new distribution of the 10 best contacts.

In Table IV, the first line represents all attack distributions for which 5 malicious peers are inserted on a single prefix, resulting in the generation of 11 different distributions (one for each prefix between 18 and 28 bits). Regarding the different repartition of peers provided by each simulated distribution, an attack is more or less efficient and easy to detect. For example, 10 malicious peers inserted to share 28 bits with the target would easily attract requests but should also trigger our detection system. At this step, we do not consider massive insertions of malicious peers to evaluate the detection mechanism. In fact, our detection mechanism only considers prefixes distribution of the 10 closest peers found after a lookup. For example, a strategy based on 15 malicious peers distributed on three prefixes like $[5 * 20; 5 * 21; 5 * 22]$ just appears as $[5 * 21; 5 * 22]$ to our detection mechanism. So, we do not need to simulate all attacks inserting more than 10 malicious peers to evaluate our detection system. Nevertheless, the countermeasure described further will deal with all malicious peers.

2) Evaluation of the K-L divergence: Finally, we evaluate the K-L divergence for these two sets of distributions, to find

Prefix	18	19	20	21	22	23	24	25	26	27	28
M (attack)	0	0	0	0	0	0	0	0	0.5	0.5	0
M (safe)	0.6	0.2	0.1	0.1	0	0	0	0	0	0	0
T	$1/2$	$1/2^2$	$1/2^3$	$1/2^4$	$1/2^5$	$1/2^6$	$1/2^7$	$1/2^8$	$1/2^9$	$1/2^{10}$	$1/2^{11}$

TABLE III: Distributions compared with K-L divergence to detect attacks

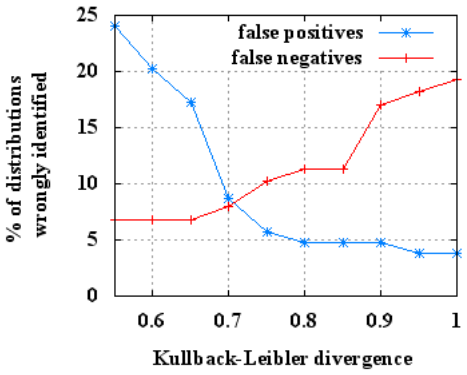

Fig. 5: False positive and false negative rates regarding the K-L divergence threshold

the proper detection threshold. The K-L divergence applied to the set of normal publications gives a mean divergence of 0.41 with a standard deviation of 0.24. To find the best threshold possible, we increase the allowed divergence while measuring the number of wrongly detected distributions. False negatives are undetected distributions from the data set of simulated attacks and false positives are safe distributions detected as attacks. Figure 5 shows that a K-L divergence of 0.7 is a good trade-off threshold to detect most DHT attacks with a good false-negative rate (7.95%) without detecting too many normal distributions (8.65%) as dangerous. With a closer look at our traces, we saw that the most dangerous attacks involving the insertion of 10 peers among the best are detected with a very small false-negative rate of 1.56%.

Attacks involving only 5 peers are harder to detect, with a false-negative rate of 21.73%, because of their smaller impact on the IDs distribution. However, such attacks involving few peers can not manage to fully control a DHT entry. Nevertheless, the system is still able to detect the most efficient configurations. A detailed analysis reveals that we mainly loose two types of inoffensive distributions:

- Attacks only targeting the 18 bits prefix. However such attacks are not efficient because they miss many requests sent to closer peers (19 bits prefix or better) and they are resource consuming because of the competition with legitimate peers at this prefix.
- Attacks with 5 peers distributed on more than 3 prefix under 23 bits. However, such attacks distributed on several prefixes are unlikely because they must rely on the absence of legitimate peers on their highest prefixes to not be detected, and they face a high competition with legitimate peers on their lowest prefixes.

Our simulations assume that legitimate peers and attackers have the same visibility through the lookup process. However the reachability of a peer can be linked to its uptime and malicious peers can have a lower churn-rate than normal peers. Although this parameter can not be taken into consideration in our simulations, this is the basic DHT function to make any contact to be found and reachable. Even if this parameter should be considered in future real experiment, it does not call our solution into question.

In summary, effective attacks that need the insertion of many peers are fully detected while undetected attacks insert not enough peers, or too far from the target, to be dangerous. The more efficient an attack is, the easier it is to detect. Now that we can detect DHT attacks by analyzing prefixes distributions, the last part presents how a peer can protect its service requests from malicious peers when an attack is detected.

C. Countermeasure

1) Progressive filtering of attacked prefixes:

Input: contact_list [ ]; prefixes_distribution [ ]; KL_increments [ ]; KL_div; max_div;
Output: updated contact_list [ ]
foreach $prefix$ *in prefixes_distribution* **do**
  KL_increments.add(partial_KL_div($prefix$));
end
KL_div = SUM(KL_increments);
while $KL\_div > max\_div$ *AND* $MAX(KL\_increments) > 0$ **do**
  prefix=KL_increments.index(MAX(KL_increments));
  remove_contacts(contact_list, prefix);
  remove_distance(KL_increments, prefix);
  KL_div=SUM(KL_increments);
end

Algorithm 1: Countermeasure to mitigate DHT attacks

Our countermeasure progressively removes peers whose prefix contributes the most to the measured K-L divergence. The procedure is described in Algorithm 1 which is executed when a distribution is detected as an attack. It provides an updated and safe contact list to send service requests. The countermeasure shares similarities with the detection mechanism: it takes place after a DHT lookup (it protects any type of service performed on the DHT) and compares the same distributions of common prefixes between 18 bits and 28 bits with the K-L divergence. Considering the first 10 best peers is sufficient to detect an attack, but not to remove all malicious contacts. The countermeasure updates and checks the 10 best peers distribution each time malicious peers are removed from the contact list. If an attacker places many peers near the target, they are progressively removed from the 10 best peers on each

iteration, until the updated distribution of the 10 best peers becomes safe, or that no prefix remains with a distribution superior to the theoretical one.

If needed, remaining contacts with lower common prefixes (17 bits or less) browsed during the lookup process are selected to complete the left places among the new list of the 10 best peers, without requesting any new lookup. This progressive approach has the advantage to avoid malicious peers while preserving the less suspicious peers among the best found. In this way, the routing efficiency does not decrease much in case of false-positives.

2) Evaluation: To evaluate our countermeasure, we used the same distributions sets than previously to estimate the detection threshold. We counted the average number of contacts removed by the countermeasure for each distribution previously detected as an attack in function of a value chosen as the maximum allowed divergence. We defined 0.7 (the same value than the detection threshold) as the more relaxed maximum divergence to consider to weaken the attack. Figure 6 shows that the procedure successfully adapts the response to the number of peers inserted. The number of removed contacts matches exactly what could be expected for the different attack cases. First, looking at the distributions set of real safe publications, false-positives distributions are just slightly impacted by countermeasures, just 2 or 3 good contacts from these distributions will be removed. In the opposite, the distributions set representing attacks with 10 malicious peers is highly affected by our countermeasure, with an average deletion between 8 and 10 malicious contacts. So, our solution succeeds to mitigate the most dangerous attacks. Finally, attack strategies based on 5 peers are also severely mitigated. In average, the countermeasure deletes between 4 and 5 malicious contacts in the corresponding distribution set.

Like the detection threshold, the maximum allowed divergence is a parameter that can be balanced to avoid false-positive distributions (with good peers) to be much affected. According to Figure 6, we consider that a maximum allowed divergence between 0 and 0.3 is good to highly mitigate attacks. If we consider the worst case, where the countermeasure removes all contacts with a 18 bits prefix or better, the published information will still remain close to the DHT entry. In fact, from our observations of safe publications, Table V indicates that the 10 best remaining peers have, in average, common prefixes between 17 bits and 15 bits. This result is far from the minimum distance of 8 bits defined to be the tolerance zone in KAD. Besides, in the worst case, the computation cost of the countermeasure keeps being negligible with a O(1) complexity: one iteration per prefix between 18 bits and 28 bits to be removed, and for each iteration, update operations are limited to scan fixed size tables.

Finally, the whole defense scheme to protect the DHT against localized attacks is presented in Figure 7. Our solution can be implemented in a future release of KAD clients, directly protecting them against DHT attacks, while keeping a full compatibility with older clients and ensuring the durability of the network. Moreover, our defense scheme can be applied as described with no overhead to every DHT whose lookup process is not delegated to intermediate peers, and on which stored information are replicated on several peers (a common solution to fight against churn).

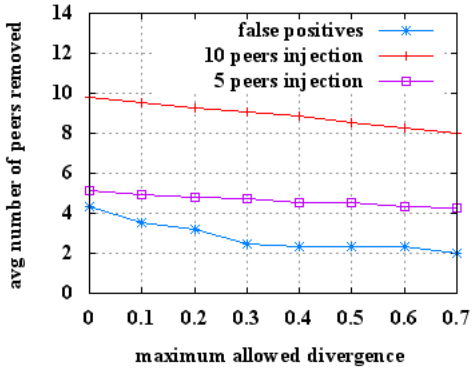

Fig. 6: Average number of contacts removed among the 10-best by the countermeasure

Prefix	Avg number of contacts
13	0.60
14	1.36
15	2.78
16	3.62
17	3.75

TABLE V: Best remaining contacts with prefix under 18bits

3) Mitigating massive malicious peers insertions: Even if our solution mitigates localized attacks, an attacker with many resources (hundreds of IP addresses) can still manage to attract many requests for the target. First, the attacker must insert many malicious peers on each prefix between 18 bits and 28 bits to make them removed by the countermeasure. Second, the attacker must overcome the number of legitimate contacts sharing a 17 bit common prefix and less with malicious ones. Such an attack can not be directly detected through a single lookup process because those IDs are too far from the target to follow the geometric distribution.

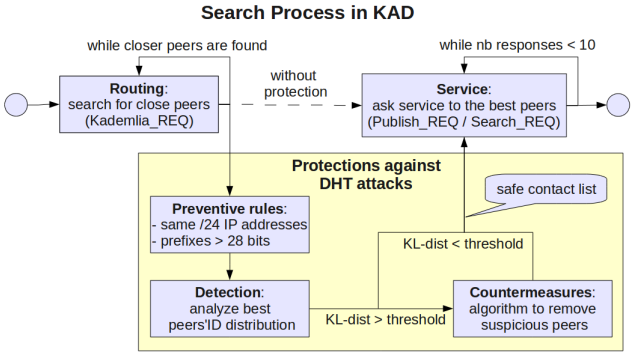

Fig. 7: Full defense scheme applied to KAD

A way to mitigate this attack, without relying on a resource consuming detection, is to stop sorting any contact found beyond a given prefix. For example, if peers sharing more than 14 bits with the target are no more sorted after the countermeasure, malicious peers will be directly in competition with hundreds of legitimate peers to attract the requests. Associated with a limitation of IP addresses, the number of distributed peers needed to to take the control of a target ID or to efficiently poison it becomes unachievable.

V. Conclusion

We presented in this paper our solution to protect the widely deployed KAD DHT against localized attacks. Our solution was designed with many constraints. We wanted a defense scheme which is efficient to detect and mitigate DHT attacks while returning a small false-positive rate. The other main constraint was to be suitable for a real implementation in the KAD network, by keeping the same DHT design and minimizing the overhead. The different parts of our defense scheme fit all these requirements.

First, our detection method is based on the comparison between two prefixes distributions: 1) real distributions of the 10 best contacts found by a peer before requesting a service; and 2) a theoretical distribution inferred from many observations of safe DHT lookups. In fact, as a first strong result, we highlighted the fact that the prefixes distribution of the best peers follows a geometric distribution $(1/2)$ because of the way legitimate peers choose their ID and the great routing efficiency of KAD. We also noticed that this distribution is not impacted by the time, the type of the published data or the distance between the publishing peer and the DHT entry. Besides the theoretical distribution, the observations also lead to the design of two preventive rules. They add new constraints on the IP address and on the ID of contacts to make attacks more difficult to perform.

We then use the Kullback-Leibler divergence as an efficient metric to detect peers' ID distribution resulting from an attack. Simulations executed on a real safe distribution set and on a second set, describing different attack strategies, showed low false-positive and false-negative rates, specially for the most dangerous attacks (1.56%), for which all the most efficient strategies are detected. Effective attacks that need the insertion of many peers are fully detected while undetected attacks do not insert enough peers to be dangerous. Finally, we described and evaluated a countermeasure to mitigate the detected attacks. It is based on a progressive filtering of contacts whose prefix distribution is the most suspicious. The evaluation showed that false-positives are not highly affected by the countermeasure while nearly all malicious contacts from the different attack strategies are removed to provide a safe DHT access. While being very efficient, our solution introduces no network overhead and a negligible computation overhead when computing the K-L divergence.

Our solution can be implemented very quickly in KAD clients by only modifying few parts in the *Search* process. Our future works will consist in implementing and testing it against attacks on the real KAD network. Such experiments will allow us to consider complex parameters like the impact of the lower churn-rate of malicious peers which could increase their visibility. Currently, the prefixes observed to detect an attack are fixed (for 4 millions nodes), and must be updated through clients if the network size significantly fluctuates. To solve this limitation and improve the efficiency of our solution, we will consider a way to evaluate dynamically the number of on-line peers in KAD. An estimation of the network size can be obtained by analyzing the routing table of a peer or by performing several DHT lookups on random entries. Then, the estimation of the network size will allow us to define dynamically what prefixes to consider to detect attacks.

References

[1] T. Cholez, I. Chrisment, and O. Festor. Evaluation of Sybil Attacks Protection Schemes in KAD. In *3rd International Conference on Autonomous Infrastructure, Management and Security - AIMS 2009*, volume 5637, pages 70–82, Enschede Netherlands, 2009. Springer.

[2] T. Cholez, I. Chrisment, and O. Festor. HAMACK: a Honeynet Architecture against MAlicious Contents in KAD. Research Report RR-6994, INRIA, 2009.

[3] G. Danezis, C. Lesniewski-laas, M. Frans Kaashoek, and R. Anderson. Sybil-resistant dht routing. In *In ESORICS*, pages 305–318. Springer, 2005.

[4] J. Dinger and H. Hartenstein. Defending the sybil attack in p2p networks: Taxonomy, challenges, and a proposal for self-registration. In *ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security*, pages 756–763, Washington, DC, USA, 2006. IEEE Computer Society.

[5] J. R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag.

[6] M. Kohnen, M. Leske, and E. P. Rathgeb. Conducting and optimizing eclipse attacks in the kad peer-to-peer network. In *NETWORKING '09*, pages 104–116, Berlin, Heidelberg, 2009. Springer-Verlag.

[7] F. Lesueur, L. Me, and V. Viet Triem Tong. A sybil-resistant admission control coupling SybilGuard with distributed certification. In *Proceedings of the 4th International Workshop on Collaborative Peer-to-Peer Systems (COPS)*, Rome, jun 2008. IEEE Computer Society.

[8] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 53–65, London, UK, 2002. Springer-Verlag.

[9] G. Memon, R. Rejaie, Y. Guo, and D. Stutzbach. Large-scale monitoring of dht traffic. In *IPTPS '09: 8th International Workshop on Peer-to-Peer Systems*, Boston, MA, April 2009.

[10] H. Rowaihy, W. Enck, P. Mcdaniel, and T. La Porta. Limiting sybil attacks in structured p2p networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 2596–2600, 2007.

[11] A. Singh, M. Castro, P. Druschel, and A. Rowstron. Defending against eclipse attacks on overlay networks. In *EW11: Proceedings of the 11th workshop on ACM SIGOPS European workshop*, page 21, New York, NY, USA, 2004. ACM.

[12] R. R. Sokal and F. J. Rohlf. *Biometry: the principles and practice of statistics in biological research*. New York: Freeman, 3rd edition, 1994.

[13] M. Steiner, T. En Najjary, and E. W. Biersack. A global view of KAD. In *IMC 2007, Internet Measurement Conference, October 23-26, 2007, San Diego, USA*, Oct 2007.

[14] M. Steiner, T. En Najjary, and E.W. Biersack. Exploiting KAD: possible uses and misuses. *Computer communications review, Volume 37 N5*, oct 2007.

[15] P. Wang, J. Tyra, E. Chan-Tin, T. Malchow, D. Foo Kune, N. Hopper, and Y. Kim. Attacking the kad network. In *SecureComm 2008: the 4th International Confere on Security and Privacy in Communication Networks*, Istanbul, Turkey, 2008. ACM.

[16] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. Sybilguard: defending against sybil attacks via social networks. *SIGCOMM Comput.Com.Rev.*, 36(4):267–278, 2006.