Detection and mitigation of localized attacks in a widely deployed P2P network

Thibault Cholez, Isabelle Chrisment, Olivier Festor, Guillaume Doyen

Springer Peer-to-Peer Networking and Applications Volume 6, Issue 2
pp 155-174, DOI 10.1007/s12083-012-0137-7

Detection and mitigation of localized attacks in a widely deployed P2P network

Thibault Cholez · Isabelle Chrisment · Olivier Festor · Guillaume Doyen

Received: 26 July 2011 / Accepted: 17 April 2012 / Published online: 10 May 2012

Abstract Several large scale P2P networks operating on the Internet are based on a Distributed Hash Table. These networks offer valuable services, but they all suffer from a critical issue allowing malicious nodes to be inserted in specific places on the DHT for undesirable purposes (monitoring, distributed denial of service, pollution, etc.). While several attacks and attack scenarios have been documented, few studies have measured the actual deployment of such attacks and none of the documented countermeasures have been tested for compatibility with an already deployed network. In this article, we focus on the KAD network. Based on large scale monitoring campaigns, we show that the world-wide deployed KAD network suffers large number of suspicious insertions around shared contents and we quantify them. To cope with these peers, we propose a new efficient protection algorithm based on analyzing the distribution of the peers' ID found around an entry after a DHT lookup. We evaluate our solution and show that it detects the most efficient configurations of inserted peers with a very small false-negative rate, and that the countermeasures successfully filter almost all the suspicious peers. We demonstrate the direct applicability of our approach by implementing and testing our solution in real P2P networks.

The final publication is available at `http://link.springer.com/article/10.1007%2Fs12083-012-0137-7`.

Keywords P2P networks · Distributed Hash Table · KAD · monitoring · security · Sybil attack · attack detection · defense

Thibault Cholez · Guillaume Doyen
Université de Technologie de Troyes, 12 rue Marie Curie - 10010 Troyes
Tel.: +33 325 715 676
E-mail: firstname.name@utt.fr

Isabelle Chrisment · Olivier Festor
LORIA, INRIA Nancy-Grand Est, Campus Scientifique - 54506 Vandoeuvre-les-Nancy
Tel.: +33 383 592 017
E-mail: firstname.name@loria.fr

1 Introduction

P2P networks, and particularly Distributed Hash Tables (DHT), represent today a major share of the usage of the Internet involving several millions of users. Their scalability, robustness and small infrastructure costs make them a perfect architecture to support file sharing applications, among others. Fully distributed P2P networks based on Kademlia, like KAD or BitTorrent's Mainline DHT, have proven being capable to handle millions of connected peers.

Malicious peers behaving badly coupled with the absence of authority can seriously disturb and damage the services offered by such P2P systems. One of the major issues affecting large open P2P networks is the Sybil attack that inserts some peers controlled by a malicious entity at a specific location inside the DHT in order to take control of an entry. Such control over the P2P network lookup mechanism allows the attacker to perform many operations like monitoring the P2P traffic [SENB07a] [MRGS09] [CCF10b], polluting the content indexation [LNR06] [LMSW10] [CCF10b], eclipsing any content [SENB07a] or launching Distributed Denial of Service attacks [NR06] [SENB07a] [WTCT$^+$08]. Several attacks affecting DHTs are well described by several researchers (the Sybil attack and its localized version), but no study has ever tried to actually detect those attacks in a real system. Concerning the mitigation of such attacks, many solutions have been proposed but none of them is suitable to protect an already deployed P2P network. In fact, the different modifications proposed to obtain a safer identity management system need deep modifications of the P2P system design and break the backward compatibility between the protected clients and the others. For instance, enforcing the placement of peers in the DHT by using a hash of their IP address is impossible in the short run: the new ID scheme would result in a trustworthy network with the new clients on one side, and an untrustworthy network with older clients on the other side. Only a drop of the backward compatibility would make the solution efficient.

We propose in this article a novel and efficient approach to detect suspicious peer insertions in a widely deployed DHT and to mitigate the effects of their possible deviant activities. We quantify the suspicious peers present in the network through large scale explorations with a very accurate crawler and we analyze the position of peers to detect those that seem to collude or to target a DHT entry. These experiments highlight for the first time, a large number of suspicious peers affecting the P2P network. In a second step, we propose a solution to protect regular peers from these deviant activities (either monitoring activities or network attacks). Our solution is lightweight and efficient. It detects the suspicious peers based on the abnormal peers' ID distribution around the targeted entry. We design a countermeasure to this local detection, preventing malicious peers from receiving requests from the protected client. Besides being able to detect and mitigate attacks, our solution involves no network overhead and fits all the constraints of already deployed P2P networks based on a DHT. We prove its direct applicability through implementations and tests performed on real P2P networks.

This article is structured as follows. Section 2 contains the related work on KAD, including both the monitoring and the security of this network, and describes the solutions proposed against the Sybil attack. Section 3 presents our crawler and our analysis of the collected data to quantify the suspicious peers in this network. Section 4 describes and evaluates our solution to detect localized

DHT attacks when a peer uses the DHT and the countermeasure to protect peers from suspicious nodes. Finally, Section 5 concludes the article and outlines our future work.

2 Related work

2.1 The KAD DHT

KAD is a widely deployed (on average ∼3 million concurrent on-line users) P2P network based on the Kademlia distributed hash table routing protocol [MM02]. It is implemented by the eMule and aMule clients which are used for file sharing.

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 thanks to the lookup process. This routing is done in an iterative way with parallel lookups by using *KADEMLIA_REQ* requests to discover new peers closer to the targeted DHT entry 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. Each KAD node has a random 128 bits "KADID" determining its position in the DHT and the references of which it is in charge of. When sharing a file, the raw data and all the keywords associated with its name are hashed separately with a MD4 function generating an ID which is then published into the DHT (to simplify our example in Figure 1, we consider the KADIDs 32, 54 and 87, respectively for each keyword and the file). The peers indexing a file or a keyword are the 10 closest peers found by the publisher around the DHT entry after a DHT lookup. The double indexing scheme allows a peer to retrieve sources for a file, from 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 publish request for a given resource, a peer is in charge of indexing this specific content and must answer any related search requests.

2.2 DHT security issues

The main well-known security issue of DHTs is the Sybil attack. As described by Douceur [Dou02], 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 [SENB07a]. They

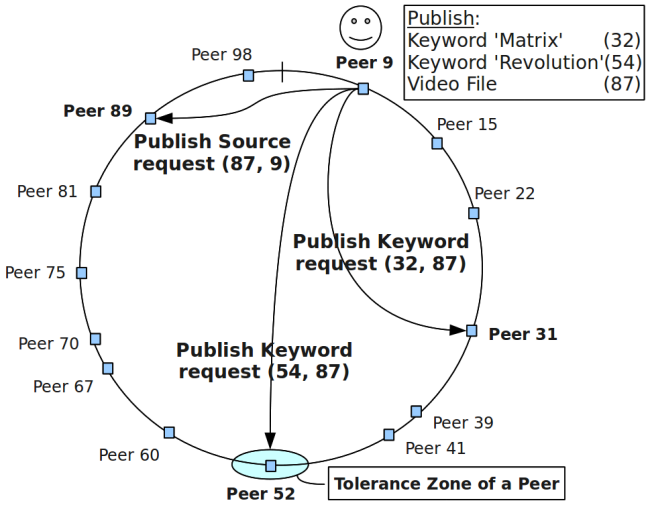

Fig. 1 KAD's double indexing scheme

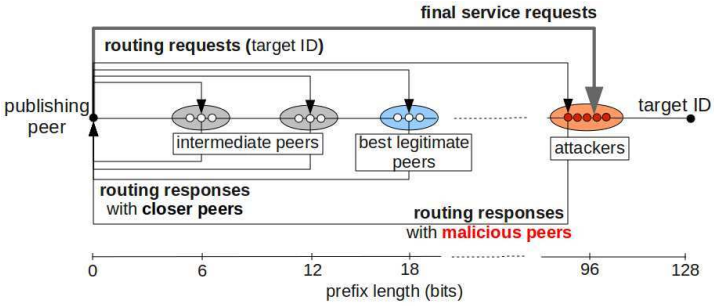

Fig. 2 Message exchange in a *Search* process under attack

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 removing some keywords indexed on the Sybils from the DHT. Another security threat of KAD consisted in overwriting the IP address of legitimate contacts in peers' routing table [WTCT+08]. The KADID spoofing exploited on a large scale allowed attackers to partition the DHT and to do massive denial of service attacks with few resources.

However, these two attacks were mitigated in the latest versions of KAD clients [CCF09]. 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 sub-network), and 3) a three-way handshake prevents a peer from overwriting the IP address of another in routing tables by claiming the same KADID. Classical Sybil attacks [SENB07a] on KAD which directly inject Sybils in routing tables have become inefficient.

Table 1 Example of peers' ID

Type	KADID	prefix
DHT entry	477221265829086C74988C40EFE63DAF	-
attacker	477221265829086C74988C4070D6E0F1	96 bits
regular peer	477229E3D7CFC729F337ABBB69C983C6	20 bits

While protective rules have been set to protect the routing table from Sybils, the lookup process remains unprotected. In fact, to control or disturb the indexation mechanism, [CCF10b] and [KLR09] adopt different strategies focusing on the *Search* process instead of the peers' routing table. In [CCF10b], A few distributed peers (with different IP addresses) are placed closer than any legitimate peer to the targeted DHT entry as shown in Figure 2 and Table 1, where the attackers share a 96 bit prefix with the target. We define the "prefix" shared between two KADIDs as the number of the first consecutive bits in common between the two KADIDs. The higher the prefix, the closer the KADIDs on the DHT are. To ensure that all inserted peers are found around the target and attract all the service requests, they cooperate to promote each other during the lookup process. Another strategy to conduct eclipse attacks is proposed in [KLR09] by generating Sybils progressively closer to the target until the lookup process stops with a timeout. For monitoring purposes, the authors of [MRGS09] insert probes in the network and limit their visibility to their closest neighbor to get a copy of the traffic sent to every peer in the DHT.

When Sybils are successfully inserted, the attacker can disturb the network in several ways. [SENB07a] presents an eclipse attack removing the indexation of the targeted contents from the DHT, [NR06] describes a way to perform a DDoS by inserting the victim's IP address in responses sent by the Sybils and can easily generate more than 100Mbit/s toward the target, [LNR06] highlights the fact that the Sybils are used to pollute the contents shared in the Overnet P2P network by announcing fake files and [LMSW10] confirms that this issue could also affect KAD.

2.3 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 [DH06] or on a trusted central authority. In [SCDR04], the authors propose to enforce 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 systems, 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 [REMP07] or an underlying social network [YKGF06]. The most successfully completed protection against the Sybil attack [LMT08] uses distributed certification coupled with a social network, but no studies so far apply its protection in a real network. Without limiting the number of Sybils, [DLLKA05] improves 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. These constraints limit the possible modifications of the P2P system and make the design of a defense scheme very challenging.

2.4 Previous explorations of KAD

A crawler is a tool that can discover all the peers participating in a P2P network and store their information. Several research papers used crawlers on KAD. The authors of [WTCT+08] and [SENB07a] perform crawls of the network in order to insert Sybils. To each peer found, they send many routing requests (*Kademlia request*) with specific KADIDs set as parameter to discover all the contacts in the peer's routing table. The crawler of [WTCT+08], called *Blizzard* has also been used in [SENB07b] to monitor the evolution of peers in time.

The authors of [YFX+09] use another algorithm to crawl the network based on bootstrap requests. They claim that this approach is more efficient (20 contacts are returned by a bootstrap request instead of 11 for a routing request) but the contacts obtained in this way are randomly chosen in the routing table which makes a precise exploration of the tables impossible. By analyzing the crawler's results, the authors highlight a large number of peers (20%) that share their KADID with another.

Even if many explorations of KAD have been conducted, none of them consider the detection of deviant activities affecting the DHT. Moreover, crawlers' accuracy is unknown and their crawling strategy is not clearly described.

2.5 Content monitoring in P2P networks

Before highlighting the monitoring activities affecting the KAD network, we remind that such activities have been recently studied in other P2P networks, mainly BitTorrent. First of all, Piatek et al. [PKK08] inserted some IP addresses of connected devices (routers and printers) inside BitTorrent's swarms but without participating in any file transfer. As a result, they received DMCA[1] take-down notices showing that the monitoring of P2P networks blindly following copyright enforcement policies, is simplistic, inaccurate and prone to false positives. The authors of [SPR09] go further in the detection of BitTorrent monitors by detecting deviant behaviors with simple heuristics (blocking connectivity, multiple clients per IP, etc.) applied on empirical observations of many popular torrents. In particular, they show that the majority of deviant peers belong to copyright enforcement policies but also that 39% are part of Botnets. Their work leads to the creation of

[1] Digital Millenium Copyright Act

more dynamic and accurate IP blacklists of deviant clients but the heavy monitoring needs make a direct application difficult in a distributed and untrustworthy environment. The authors of Omnify [PSFNR11] choose another strategy to detect monitors based on the construction and the investigation of a reverse infohash database that unifies the peers sharing a same content. The analysis of the collected IP addresses shows that monitors can be detected as they participate to several swarms of the same content, contrary to regular peers. However, monitors still miss a significant part of torrents. Finally, Le Blond et al. [LBLLF+10] make a very efficient monitoring architecture for BitTorrent that pro-actively monitors the new torrents. Thanks to a good knowledge of the protocol and an exhaustive gathering of torrents' public data, they are able to identify 70% of the new content providers with a single computer which proves that BitTorrent's central tracking system can ease a lot the task of monitors. Compared to these studies, we show that such deviant peers also affect the KAD network and we provide a distributed and efficient way to detect them, which does not imply a heavy monitoring activity.

3 Quantifying suspicious peers in the whole network

In this first section, we present our crawler which is able to build an accurate view of the KAD network. Then, we analyze the collected data to quantify, for the first time, suspicious peers inserted on the DHT.

3.1 Crawling the KAD DHT

3.1.1 Crawling approach

Our crawler must achieve two goals: it must get a very accurate view of the network and limit the footprint of the crawl by limiting the number of requests sent to each peer. Sending few requests makes the crawl compatible with the recent flooding protection that limits the number of requests received by a peer in a short period of time from a given source [CCF09]. Our crawling strategy involves three steps as follows.

Bootstrap: In the first step, *Bootstrap* requests are sent to get quickly a first, yet inaccurate, view of the DHT. For each request sent, the crawler receives 20 contacts randomly chosen in the peer's routing table. The new contacts retrieved from the *Bootstrap* responses are progressively requested until 500000 contacts are found by the crawler with at least 500 contacts per zone[2]. Beyond this value, the contacts returned by *Bootstrap* requests being randomly chosen, new contacts are more difficult to find with this method.

Full crawl: The second step consists in crawling with accuracy each zone thanks to the lookup requests (*Kademlia* request). Each requested peer returns the 4 closest contacts to the KADID given as parameter that are stored in its routing

[2] a zone is an artificial subdivision of the DHT address space considering only the first byte of the KADID (from *0x00* to *0xFF*)

table. To discover all the peers, we generate $2^{21} \approx 2$ million KADIDs we define as landmarks and that are uniformly distributed over the DHT space. We create for each landmark a lookup request and send it to the closest peer already discovered.

Second crawl: When a zone has been fully explored, that is to say when all landmarks have been sent, the final step consists in performing a second crawl of the zone. For each contact already known, we compute the prefix shared with its direct neighbor and we generate a new landmark placed between the two peers. Finally, a lookup request is sent to the peer. This final step is very accurate and allows the crawler to find some peers missed during the first crawl. The exploration is over when all peers have been requested about their neighbor.

3.1.2 Cartography of the DHT

Stored information: For each peer discovered by the crawler, we store the following information: `<KADID, IP address`[3]`, TCP port, UDP port, KAD protocol version, state of the peer>`. The state of the peer can be P(*possible*), T(*tried*) or R(*responded*) respectively if the contact has just been discovered, has been requested or has responded.

```
[...]
<32FFF76959F6A7095347FB338B304330, 111.161.X.X,38060,16905,0,T>
<32FFFC5C4D5AE9A082871FF68B1F0D9C, 219.139.X.X, 5149, 1025,4,R>
<32FFFC5C4D5AE9A082871FF68B1F0D9C, 219.139.X.X, 5149, 5159,4,P>
Zone 33: 15196 contacts
<3300048A90460A8AAC3DD2FF542ADF98, 86.212.X.X, 12399,39949,9,R>
<3300083A0480CFA91B8C142401DD26F2, 111.240.X.X, 5611, 5621,8,T>
<330018506569424D7CBA7133F437EDC8, 120.35.X.X,  6647, 6657,8,R>
<33002596F7AAAA4348FB4349F0A14FA4, 187.78.X.X, 46318,61632,9,R>
<33002EF905E27753B1900BC602D29C20, 218.92.X.X, 19774,19774,8,T>
<33004546934FABE9685674DE1598548F, 125.81.X.X, 51478,52073,9,R>
[...]
```

Global results: During our measurements, a zone contains between 13000 and 17000 contacts, the overall number of peers varies from 3.3 million to 4.3 million given the day and the hour of the crawl. At a large scale, the peers are uniformly distributed over the DHT address space (Figure 3) which is expected because the majority of the regular peers randomly choose their KADID when connecting for the first time to the network. We analyze more precisely the results of our explorations in the next subsection with the goal to detect the suspicious peers' positions that indicate deviant activities.

3.1.3 Evaluation of the crawler

The crawls of KAD were performed from a single computer, with each zone crawled sequentially. With the highest accuracy (2 pass), a snapshot of a zone takes 2 minutes and a full crawl of KAD takes less than 9 hours. The speed of the crawl is not a critical parameter in our research purpose, because deviant peers must be quite stable, but the accuracy of the crawler actually is. To validate our crawling

[3] some IP addresses are anonymized in the article

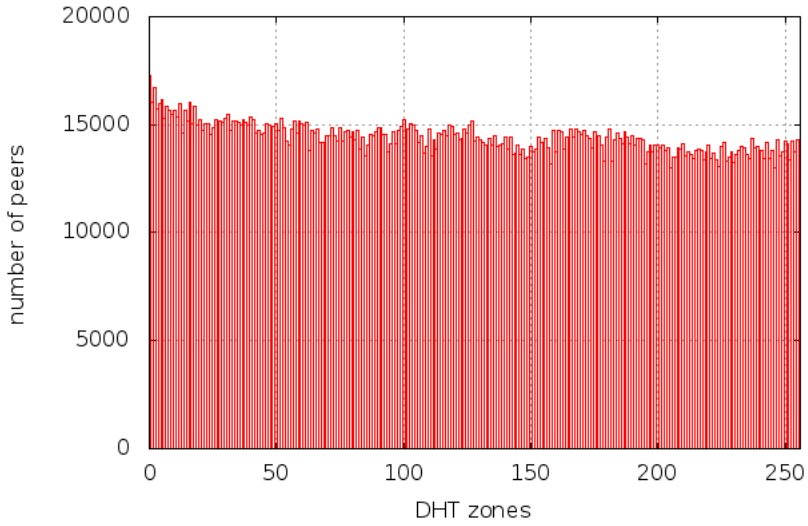

Fig. 3 Distribution of peers' ID over the zones of the DHT

approach, we evaluated our crawler efficiency with two experiments. First, we inserted 360 peers in the KAD network thanks to the PlanetLab testbed[4]. The peers follow an attack pattern: they form groups of 5 and are placed around 72 DHT entries. Each peer shares at least 96 bits with its target. At the end of the crawl, all the inserted peers were found in the crawler's data. The following sample shows the results of a data analysis looking for peers close to the 72 targets we defined.

```
[...]
KADID 71: 19856E29730F11CA0E0C210630ADCB36
<19856E29730F11CA0E0C210621142E70, 62.108.171.74, 14337, 13602, 8, T> [prefix = 99]
<19856E29730F11CA0E0C2106546F8C89, 193.167.187.186, 14690, 13799, 8, T> [prefix = 97]
<19856E29730F11CA0E0C21065622F60F, 155.245.47.241, 13953, 13779, 8, T> [prefix = 97]
<19856E29730F11CA0E0C210676E74885, 212.51.218.235, 13897, 14465, 8, T> [prefix = 97]
<19856E29730F11CA0E0C21069636476A, 129.97.74.14, 14308, 13853, 8, T> [prefix = 96]
KADID 72: EBCBA6D72037ED01F56809A9FFE6A86E
<EBCBA6D72037ED01F56809A9268DA7FB, 155.245.47.241, 13915, 13842, 8, T> [prefix = 96]
<EBCBA6D72037ED01F56809A94519B1D4, 129.97.74.14, 14029, 13914, 8, T> [prefix = 96]
<EBCBA6D72037ED01F56809A9702F72B7, 193.167.187.186, 13666, 14427, 8, T> [prefix = 96]
<EBCBA6D72037ED01F56809A9892C91A4, 62.108.171.74, 13853, 14683, 8, T> [prefix = 97]
<EBCBA6D72037ED01F56809A9BAD2A19E, 212.51.218.235, 13861, 13939, 8, R> [prefix = 97]

72/72 of the proposed KADIDs are targeted with at least 96 bits by:
37 IP addresses (showing 361 unique KADIDs in the whole crawler's data)
```

The second evaluation of our crawler consisted in instrumenting a KAD client to print the list of contacts found after a DHT lookup while the crawler explored the corresponding zone of the DHT. In all our tests, the peers found by the instrumented client were present in the crawler's data, which shows the relevance of our crawling approach.

[4] http://www.planet-lab.org/

3.2 Detection of suspicious peers in the DHT

As explained, an efficient DHT monitoring or attack involve the insertion of one or several peers close to the targeted DHT entry in order to attract the requests sent by the regular peers for that content. To improve the attack efficiency, many peers can be inserted around the DHT entry.

3.2.1 Analysis of the distances between peers

In KAD, like in most DHTs, the normal behavior of legitimate peers is to randomly choose their ID (128 bits KADID). As the distribution is uniformly random, the average distance between two IDs is only defined by the number of peers in the DHT. Let F be the function giving the mean number of peers sharing at least x bits with a given KADID, for a total number of N peers in the network.

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

More precisely, the theoretical prefixes (number of common bits) distribution of peers' ID around a target ID follows a geometric distribution with parameter $p = 1/2$.

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

Figure 4 shows the mean number of peers existing in the DHT according to the length of consecutive common bits (prefix) shared with a target ID. We consider a realistic number of 4 million peers taking part in the DHT at the same time. Table 2 presents some values of F with $N = 4 \times 10^6$ and $x \in [1; 128]$. Moreover, the average prefix shared between two peers is $d_{moy} = log_2(N) = 21.93$ bits.

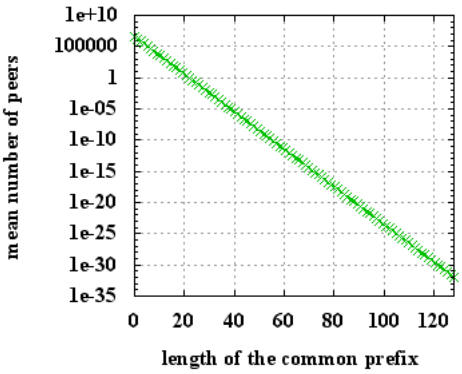

Fig. 4 Mean number of peers sharing a given common prefix size with a target

Our first analysis consisted in detecting those groups of peers which are very close to each other on the DHT because they all focus on the same DHT entry. Our assumption is that two peers sharing a very long prefix with each other have

Table 2 Mean number of regular peers sharing a given prefix with a given KADID in a 4 million peers DHT

Number of bits in common	Average number of matching peers
1	2 000 000
8	15625
12	976
16	61
18	15.25
20	3.8
24	0.24
28	0.015
32	$9.32 * 10^{-4}$
64	$2.17 * 10^{-13}$
96	$5.05 * 10^{-23}$
128	$1.17 * 10^{-32}$

an unlikely position regarding the regular uniform distribution of the KADIDs. A suspicious position can be intentional for monitoring or attack purposes.

Based on our crawl of the KAD DHT, we computed the distribution of prefixes shared between each peer and its direct neighbor which is presented in Figure 5. While a regular distribution should strictly decrease, the measured distribution clearly shows abnormalities.

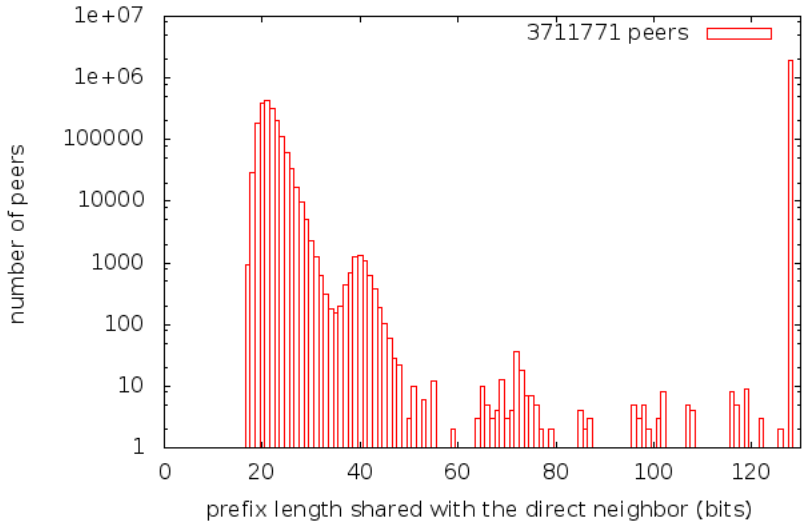

Fig. 5 Distribution of the prefixes length between neighbors on the DHT

Figure 6 shows a clear deviation from the theoretical distribution for peers sharing between 35 and 45 bits. When the shared prefix between two peers is higher, the probability of an intentional position is also higher because the prob-

ability to find two close neighbors decreases when the length of the shared prefix increases.

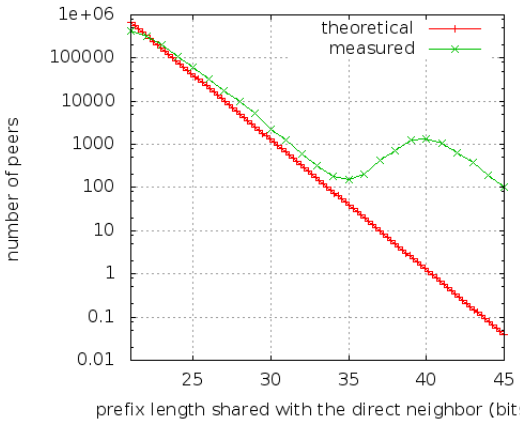

Fig. 6 Theoretical and measured mean number of peers sharing a given prefix with their neighbor

Given the number of peers in the DHT, prefixes below 30 bits are likely shared between several peers and make impossible a proper detection of attacks. However, we can say that prefixes over 35 bits are unlikely enough to indicate a very suspicious position of the peers. Considering that two peers sharing more than 35 bits are suspicious, the analysis of our crawler's data highlights 2074 groups of suspicious peers with obvious attack patterns. In fact, the suspicious peers come from the same sub-network (123.144.0.0/16 or 123.145.0.0/16) and use specific ports (10875 or 10839) to connect to the P2P network. The following sample shows two of these groups:

```
Prefix "4A9D8C8777000000000000000000000", length 40, shared by 6 contacts:
<4A9D8C87774AF8C551FE78BDDC3F5A37, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C877780DFB9985E75EE92AD1C68, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C877780DFB9985E75EE92AD1C68, 123.145.X.X, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C8777F0F03BD1FE123548E269D2, 123.144.X.X, 10839, 10839, 0, R>

Prefix "4A9D8C8777800000000000000000000", length 41, shared by 4 contacts:
<4A9D8C877780DFB9985E75EE92AD1C68, 123.145.X.X, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.X.X, 10875, 10875, 8, T>
<4A9D8C8777F0F03BD1FE123548E269D2, 123.144.X.X, 10839, 10839, 0, R>
```

However, our detection strategy has two limitations. First, as we only analyze the distances between two peers, we can not detect when a single peer is placed near a DHT entry. Second, we highlight many groups of suspicious peers but we do not know what they target. Therefore, we will present in the next subsection another detection strategy based on the analysis of the distances between peers and well-known contents.

3.2.2 Analysis of the distances between peers and contents

To apply this detection method, the first step consists in collecting keywords about contents shared in the KAD network. As input, we used several sources of multimedia contents (Amazon top sellers, iTunes, ThePirateBay) to get relevant keywords and then we computed their MD4 value that gives their KADID. Finally we computed the list of the suspicious peers that are too close to different contents (i.e. sharing a prefix > 35 bits) in the crawler's data. The following sample shows some suspicious peers found by our analysis:

```
[...]
twilight 4D62D26BB2A686195DA7078D3720F60A
<4D62D26BB2A686195DA7078D3720F632, 95.175.X.X, 7290, 7294, 8, R> [prefix = 122]
soundtrack AC213377BB53F608390BD94A6AE6DD35
<AC213377BB53F608390BD94A82582F42, X.X.X.X, 5003, 5002, 8, R> [prefix = 96]
harry 770CF5279AB34348C8FECF9672747B94
<770CF5279AB34348C8FECF96524D8CDE, X.X.X.X, 5003, 5002, 8, P> [prefix = 98]
robin B9DF47E5BFAD75F8EE5E3F50EA217983
<B9DF47E5BFAD75F8EE5E3F5051F34AA8, X.X.X.X, 5003, 5002, 8, R> [prefix = 96]
<B9DF47E5BFAD75F8EE5E3F50EA21799F, 95.175.X.X, 7290, 7294, 8, R> [prefix = 123]
[...]

216/888 of the proposed keywords are targeted with at least 96 bits by:
44 IP addresses (showing 2119 unique KADIDs in the whole crawler's data)
```

One quarter of the 888 keywords we monitored is obviously targeted by a peer sharing at least a 96-bits prefix. This prefix length clearly indicates a malicious position, given the very low probability to find such an honest peer (Table 2). A sample of the keywords that are targeted is listed in table 3. Moreover, some attack patterns appear easily from collected data. For instance, 205 of the 216 keywords are targeted by peers using the UDP port 5003 and TCP port 5002, they share a 96 bits prefix with the content's KADID and use IP addresses distributed over the Internet. Another group is focused on 16 keywords with peers using the UDP port 7290 and TCP port 7294, sharing a 122 bits prefix with each target and using IP addresses from the sub-network 95.175.0.0/16.

Table 3 Sample of targeted keywords

Keyword	closest peer
avatar	126
invictus	123
sherlock	122
princess	122
frog	98
ncis	96
nero	96
...	...

Keyword	closest peer
nine	122
love	122
american	97
russian	97
the	96
black	96
pirate	96
...	...

Considering the 44 suspicious IP addresses involved in the 216 suspicious groups, we searched in the crawler's data other peers sharing these IP addresses. In fact, the list of keywords used for the detection is just a small subset of the contents actually shared within the P2P network. It appears that the suspicious

peers we found thanks to our set of keywords only represent 10% of the peers using these 44 suspicious IP addresses. Overall, these suspicious IP addresses account for 2119 different peers in the DHT, which clearly shows that a lot of contents shared within KAD are targeted by deviant peers.

This second detection method could be extended. For instance we did not consider keywords written using Asian characters which may lead to miss many other peers as KAD is widely used in China [SENB07b]. Moreover, besides keywords, all the files indexed in the DHT can also be targeted. A tool collecting the different fileID shared in the network through the P2P system search engine could be useful to detect such activities.

4 Protecting peers from localized DHT attacks

In the previous section, we presented two complementary detection strategies and proved through large scale measurements that the KAD DHT is widely affected by thousands of peers targeting shared contents and that these peers could perform content-centric attacks. In this section, we propose to detect such suspicious peers presenting an attack configuration when a client performs DHT lookups. Part of this section was published in [CCF10a].

4.1 Analysis of peers' ID distributions for safe DHT entries

Localized DHT attacks are very efficient on KAD because the 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, we showed that a localized attack introduces proximity abnormalities in IDs distribution around the target. Therefore, to detect DHT attacks, we consider the unusual peers' ID distribution resulting from the position of these malicious peers and that we can find during the regular lookup process, without any network overhead.

4.1.1 Theoretical IDs distribution

We described in Section 3.2 the theoretical IDs distribution for a DHT. However, even if the IDs distribution is well known at the network scale (i.e. when using a crawler), 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 the lookup procedure to find the best peers on which to publish or to search contents. It is difficult to infer if the peers found after a DHT lookup are indeed the best potential peers existing in the DHT. Moreover, several parameters can affect the routing efficiency of a peer (uptime, distance to the target ID...) and need to be considered.

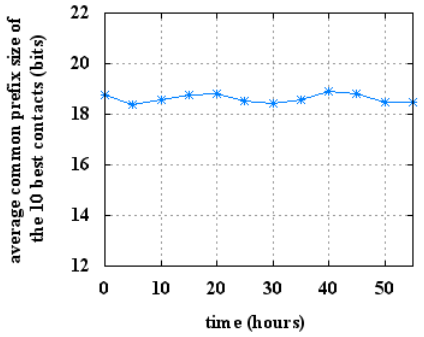

Fig. 7 Average common prefix of the 10 best contacts in time

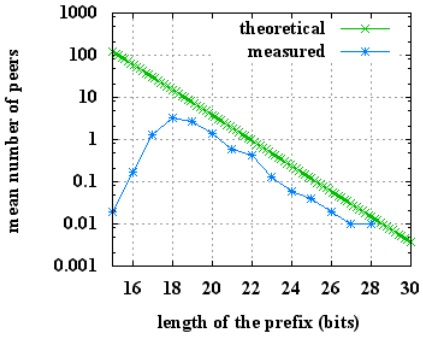

Fig. 8 Prefix distribution of the 10 best contacts

4.1.2 Real IDs distributions

To compare the results of the KAD lookup algorithm to the theoretical random ID distribution, we ran an experiment that consists in performing many lookups on safe DHT entries and storing 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 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 ($12\times100+3\times200$) 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 7 shows that the average common prefix size of the 10 best contacts found is between 18 bits and 19 bits. This 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 4). This result confirms the KAD routing efficiency since the 10 best peers found after the lookup process are nearly the 10 best peers that actually exist in the DHT. Figure 7 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 see a very small periodic variation of the average common prefix size. This small amplitude is explained because KAD is mainly used in European countries and in China. The activity hours of these two regions are complementary.

We investigated several other parameters potentially affecting the quality of the best found contacts. We showed that the DHT distance between the requesting peer and the target does not impact the routing efficiency. In the same way, the type of published information (keyword or file), or the type of requested services (publish or search) has no impact on the quality of the found contacts.

Table 4 Mean number of peers among the 10 best contacts found sharing a given common prefix size with a target

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

Finally, the most important result concerning this experiment is given in Figure 8. For each publication, we counted 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 section 3.2. So, the KAD lookup procedure is efficient enough to give a representative view of the closest peers possible. Above all, this result shows that, when the size of the DHT is well-known, the theoretical random ID distribution is sufficient to characterize the results obtained in a real lookup process.

We will rely on this strong result to detect DHT attacks. We will first define some preventive rules to avoid the most obvious insertions of peers. In a second time, we will present a metric that can accurately detect the remaining attacks.

4.1.3 Preventive rules

Recent Sybil attack protection mechanisms introduced in KAD [CCF09] demonstrated that few passive rules 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 be achieved.

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

Discarding too close nodes: A second protection consists in setting a minimum distance between the target ID and the closest responsible peers, according to

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 5 Distributions compared with K-L divergence to detect attacks

the normal peer behavior and the DHT size. As shown previously, we assume that legitimate peers randomly choose their ID. So, we 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 service requests. However, peers sharing a very high common prefix (i.e. between 28 and 128 bits) are very unlikely as illustrated in Figure 8. So, we define a minimum distance between the target ID and the closest responsible peers of 28 bits, which moves 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.

4.2 Detection and protection against localized attacks

4.2.1 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 important to consider because they receive the service requests which hacking is the root of an attack.

The main issue is that the few best peers represent 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, 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 [SR94]. Given the probability distributions M and T of a discrete random variable, the K-L divergence between M and T is defined as:

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

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 5 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 4 that focusing on the 17 bits prefix or less would only permit an attacker to capture on average 1.26 out of the 10 requests, while being in competition with many legitimate peers. Second, as seen in Figure 8, the average distribution of the best contacts currently observed in KAD follows the geometric distribution (1/2) from the 18 bits prefix. This limit 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. This means that by design, our detection algorithm needs to know the size of the network, or at least, what is the regular distribution. We present in Section 4.2.4 how a peer can easily learn this information. Concerning the other limit 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 positive detection. 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 and must be defined as the detection threshold.

4.2.2 Evaluation with simulated attacks distributions

To define the 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 4.1.2. The second set of distributions is simulated to represent different attack scenarios.

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 explainedearlier, our detection mechanism considers prefixes distribution of the 10 best peers sharing a common prefix between 18 bits and 28 bits. We consider that all inserted malicious peers are found during the lookup process.

To simulate attacks distributions, we first initialize a distribution with, for each prefix length in the detection window [18-28], 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 insertion strategies that use between 5 and 10 malicious peers, and we finally compute the new distribution of the 10 best contacts that includes the malicious ones. Table 6 presents the different attack

Table 6 Possible repartition of peers with common prefixes between 18 & 28 bits

# of malicious peers inserted	# of prefixes used	Repartition of the peers	# of simulated attacks
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
5	1	5	11
5	3	2-2-1	8
5	5	1-1-1-1-1	6

configurations we considered from the most obvious inserting 10 peers on the same prefix (11 different attacks are simulated, one for each prefix between 18 and 28 bits where the 10 peers can be inserted), to the least visible inserting 5 peers on different prefixes. 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 all requests but would also be easily detected.

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 inserted malicious peers.

Evaluation of the K-L divergence

Finally, we evaluate the K-L divergence for these two sets of distributions, to find 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 9 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 see that the most dangerous attacks involving the insertion of 10 peers 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. Nev-

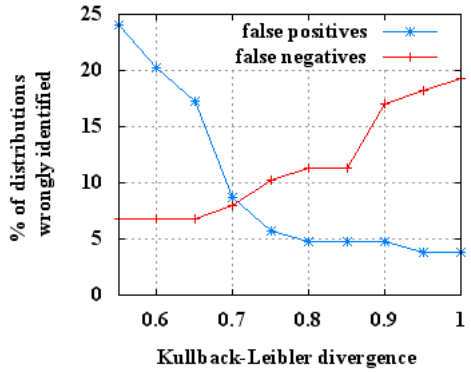

Fig. 9 False positive and false negative rates regarding the K-L divergence threshold

ertheless, the system is still able to detect the most efficient configurations. A detailed analysis reveals that we mainly lose 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 in order to not be detected, and they face a high competition with legitimate peers on their lowest prefixes.

In summary, effective attacks using 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. After having shown that we can detect DHT attacks by analyzing prefixes distributions, we present, in the next part, how a peer can be protected from malicious peers when an attack is detected.

4.2.3 Countermeasure

Progressive filtering of attacked prefixes

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 to.

```
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

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, (i.e. the updated KL-divergence is under the maximum allowed divergence "*max_div*"), or that no prefix remains with a distribution superior to the theoretical one.

This progressive approach has the advantage of avoiding malicious peers while preserving the less suspicious peers among the best found. 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. In this way, the routing efficiency does not decrease much in case of false-positives and an attacker could not eclipse a content by triggering on purpose our countermeasure to reduce the replication factor of a content, the 10 replicas being always granted after the countermeasure.

Finally, the whole defense scheme to protect the DHT against localized attacks is presented in Figure 10. 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).

Evaluation by simulation

To evaluate our countermeasure, we used the previously described distributions obtained from regular lookups and simulated attacks. 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

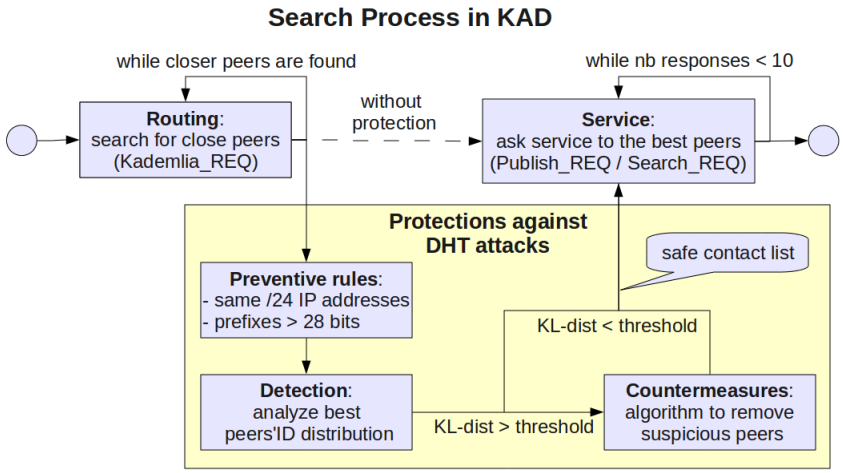

Fig. 10 Full defense scheme applied to KAD

Table 7 Best remaining contacts with prefix under 18bits

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

divergence. We defined 0.7 (the same value than the detection threshold) as the more relaxed maximum divergence to consider. Figure 11 shows that the countermeasure 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 scenarios. First, looking at the distributions of real safe publications (false-positives), they are just slightly impacted by the countermeasure; just 2 or 3 good contacts from these distributions will be removed. In the opposite, the distributions representing attacks with 10 malicious peers are highly affected by our countermeasure: between 8 and 10 malicious contacts are removed. 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 to be much affected. According to Figure 11, 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 must remove all contacts in the detection window, the published information will still remain close to the DHT entry. In fact, from our observations of safe publications, Table 7 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 as the tolerance zone in KAD.

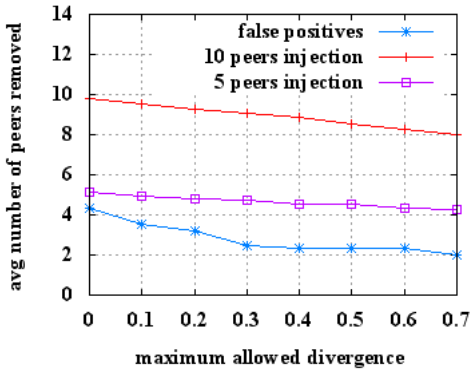

Fig. 11 Average number of contacts removed among the 10-best by the countermeasure

Evaluation on PlanetLab

To prove the applicability and the efficiency of our approach, we implemented our protection in an aMule client and monitored it while accessing 20 keywords we attacked from PlanetLab. The attacks consisted in the deployment of 100 modified clients from 10 PlanetLab nodes. These clients were modified to choose their ID close to a specified keyword. We made the attacks hard to detect by only involving 5 Sybils per attack and we placed them to share at least 20 bits with each targeted keyword. We show in Figure 12 a snapshot of the peers found after a lookup on the attacked keyword "madonna".

```
Detection results:
Name: madonna
Hash: A35BC8A4D252ADB3A99A46A28B275DFB

Responded: 26 21 21 21 20 20 19 18 18 17 16 16 11
Prefix between 17 and 28 from the 10 best distributions: 10 (13 total)
Ratio (responded[17-28]/responded): 0,77

KL distance: 1.093836, details:
 KL increments  -0,13 -0,13 -0,02 0,23 0,68 0,00 0,00 0,00 0,00 0,46 0,00 0,00
     Prefixes     17   18   19   20   21   22   23   24   25   26   27   28
       M        0,10 0,20 0,10 0,20 0,30 0,00 0,00 0,00 0,00 0,10 0,00 0,00
       T        0,38 0,38 0,12 0,06 0,03 0,02 0,01 0,00 0,00 0,00 0,00 0,00
```

Fig. 12 Detection of a localized attack on the keyword "madonna"

The numbers highlighted in blue are the prefixes of the different responding peers found during the lookup process (including our Sybils): 13 were found and the 10 closest are used to compute the measured distribution M. The attack is detected because the Kullback-Leibler divergence computed for this distribution is above the threshold ($1.09 > 0.7$).

As a consequence, the countermeasure is applied and the resulting distribution is presented in Figure 13. We can see that 4 out the 5 Sybils were filtered by our

```
Countermeasure results:
Name: madonna
Hash: A35BC8A4D252ADB3A99A46A28B275DFB

Responded: 20 20 19 18 18 17 16 16 11
Prefix between 17 and 28 from the 10 best distributions: 6 (9 total)
Ratio (responded[17-28]/responded): 0,67

KL distance: 0,431523, details:
 KL increments  -0,14 -0,04 0,05 0,56 0,00 0,00 0,00 0,00 0,00 0,00 0,00 0,00
     Prefixes     17   18   19   20   21   22   23   24   25   26   27   28
       M        0,17 0,33 0,17 0,33 0,00 0,00 0,00 0,00 0,00 0,00 0,00 0,00
       T        0,38 0,38 0,12 0,06 0,03 0,02 0,01 0,00 0,00 0,00 0,00 0,00
```

Fig. 13 Countermeasure applied to mitigate the attack on the keyword "madonna"

countermeasure (on the prefixes 26 and 21). The maximum allowed divergence "*max_div*" of the countermeasure were set to 0.7 for this experiment and we can see that the final KL divergence is below this threshold ($D_{KL} = 0.431523$). A lower stopping threshold (for example set to 0) would have removed the peers with the 20 bits prefix, filtering the last Sybil but also a legitimate peer. We still recommend the developers to set a stopping threshold of 0 in order to avoid as much Sybils as possible. Considering the other keywords, all attacks were detected and between 4 and 5 malicious peers were removed which is exactly what was expected from our simulations.

4.2.4 Design improvements

Feedback on implementations

To prove that our solution is suitable to protect currently deployed P2P networks based on a DHT, we implemented it on KAD. The main developer of the gtk-gnutella client[5], Raphael Manfredi, also implemented our solution in his client to protect the Gnutella DHT. Thanks to his work we got a feedback on the applicability of our solution to other P2P networks and we got the opportunity to improve some points to make our solution easier to deploy. Both implementations of the detection and countermeasure were straightforward and just one source file per client needed modifications (*Search.cpp* for KAD, *lookup.c* for gtk-gnutella). However the detection parameters are specific to each network.

gtk-gnutella: Two main differences between KAD and the Gnutella DHT make the detection parameters set for KAD inefficient. In particular, the prefixes [18-28] observed by our solution to detect attacks are not suitable to this network. First, the Gnutella DHT is much smaller than KAD with approximately a population of $7 * 10^4$ peers. Second, information published on the Gnutella DHT is replicated on 20 peers while it is replicated on 10 peers in KAD. The following execution shows a proper detection of suspicious peers on the Gnutella DHT. We can see that the 20 peers found after a DHT lookup share lower prefixes [11-16] with the target.

[5] `https://gtk-gnutella.svn.sourceforge.net/svnroot/gtk-gnutella/trunk/gtk-gnutella/`

```
11bit prefix: M=0.50 (10/20 nodes,log2=-1.00000) T=0.500000 => K-L contrib: 0.000000
12bit prefix: M=0.20 (4/20 nodes,log2=-2.321928) T=0.250000 => K-L contrib:-0.064386
13bit prefix: M=0.15 (3/20 nodes,log2=-2.736966) T=0.125000 => K-L contrib: 0.039455
14bit prefix: M=0.05 (1/20 node, log2=-4.321928) T=0.062500 => K-L contrib:-0.016096
15bit prefix: M=0.05 (1/20 node, log2=-4.321928) T=0.031250 => K-L contrib: 0.033904
16bit prefix: M=0.05 (1/20 node, log2=-4.321928) T=0.015625 => K-L contrib: 0.083904
with 20/20 nodes, K-L divergence to bce4d59bd8db868b7ffc0031ae81cca8db51937f = 0.076780
```

To be adapted to the Gnutella DHT, the prefixes considered by our solution to detect the suspicious peers have to be lowered. This highlights the need of an automatic way to compute the right prefixes used for the detection window in order to adapt easily our solution to different networks. Moreover, the population of a P2P network can quickly vary in time. For example, the size of the Gnutella DHT was divided by 10 when the LimeWire client was shutdown. Such a variation in the number of peers would make our detection inefficient if no process can adapt the detection parameters accordingly. A quick decrease of the number of peers would leave space for attackers to be inserted without triggering the detection while a quick increase would always trigger the countermeasure. A last argument in favor of dynamic detection parameters is the daily variation the network population. KAD does not suffer from this problem because its population is well balanced between Europe and China. However, we present in Figure 14 the results of a previous experiment [TCCF11] measuring the size of two DHTs during a few days and they show that the population of BitTorrent's Mainline DHT can be divided by two during a day. A periodic computation of the detection window would improve the accuracy of our detection.

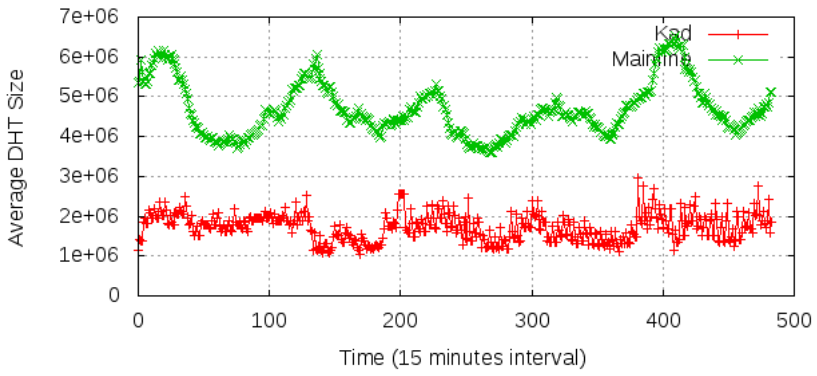

Fig. 14 Evolution of the population for two DHTs in time

Dynamic setting of the detection window

Two approaches are possible to dynamically compute the detection window regarding if the DHT size is already known or not.

Using the DHT size: Many clients of P2P networks (eMule, Vuze, gtk-gnutella, etc.) provide an estimation of the size of the network based on the analysis of

their routing table. The setting of a proper detection window is straightforward with the knowledge of the DHT size. Let N be the DHT size given by the client and K the replication factor used on the DHT. The first prefix considered by the detection window is given by the number of bits "bmin" defined by:

$$bmin = \lfloor log2(N \div K) \rfloor \tag{4}$$

The $\lfloor x \rfloor$ notation stands for the integer part of the number because the number of bits must be an integer. When applied to KAD and Gnutella DHT, the equation gives the right prefixes to start the detection window, respectively 18 bits and 11 bits (for KAD: $bmin = \lfloor log2(4 * 10^6 \div 10) \rfloor = 18$ and for gtk-gnutella: $bmin = \lfloor log2(7 * 10^4 \div 20) \rfloor = 11$). The high limit of the detection window "bmax" must be defined so that peers beyond that limit are too close from the target and automatically filtered. We chose $bmax = bmin + \delta$ with $\delta = 10$ to limit the number of false positives of peers directly filtered to $1/2^{10}$ which is less than 0.1%.

Learning of the detection window: If the client of the P2P network does not provide an estimation of the DHT size, the detection window and the regular distribution of prefixes can easily be learned from the network. The peer must perform several DHT lookups for randomly generated KADIDs and then compute the average distribution of the found prefixes. Algorithm 2 describes this learning method.

```
Input: nb_lookup
Output: prefixes_distribution [ ]
nb_contacts=0;
for i=0; i<nb_lookup; i++ do
    rdm_id = generate_random_id();
    10_best_contacts [ ] = lookup(rdm_id);
    foreach contact in 10_best_contacts do
        current_prefix = compute_prefix(rdm_id, contact);
        prefixes_distribution [current_prefix] ++;
        nb_contacts ++;
    end
end
foreach prefix in prefixes_distribution do
    prefix = prefix / nb_contacts;
end
```

Algorithm 2: Algorithm to learn the regular distribution of prefixes from DHT lookups

5 Conclusion

We have presented an integration solution to quantify, detect and mitigate the localized insertions of suspicious peers in P2P networks based on a DHT. The first contribution consisted in quantifying all suspicious peers present at the network scale. To this end, we built an accurate crawler implementing a novel strategy. Its efficiency was demonstrated on a real network. We then analyzed the collected data. The analysis led to the design of two algorithms to detect suspicious peers on

DHTs. The first one considers the distances between peers which highlights 2074 groups of suspicious peers surrounding an unknown target in the KAD network while the second algorithm considers the distances between peers and some predefined contents which highlights 2119 other deviant activities based on a single peer insertion very close to the content. The two detection methods are complementary and highlight different types of threat from the crawler's data. For the first time, we proved that a widely deployed DHT is actually targeted by the insertion of Sybils which may lead to major security issues for the regular peers (monitoring, pollution, eclipse attack, DDoS, etc.).

Having witnessed this real threat, we designed an advanced yet efficient scheme to protect the widely deployed KAD DHT against localized attacks. Our solution was designed with many constraints in order to be efficient and able to fix an already deployed network. We wanted a defense scheme that can mitigate DHT attacks while returning a small false-positive rate, keeping the same DHT design and minimizing the overhead. The different parts of our defense scheme fit all these requirements. Our detection method is based on the comparison between the theoretical and the real distribution of prefixes shared between the closest contacts found near a target after a DHT lookup. The theoretical distribution was inferred from many observations of safe DHT lookups and we proved that the closest peers' prefixes follow a geometric distribution (1/2). Then, we used the Kullback-Leibler divergence as an efficient metric to detect peers' ID distribution resulting from an attack. Our evaluation based of simulated attacks shows low false-positive and false-negative rates, especially for the most dangerous attacks. The more efficient the attack, the better is its detection. 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. Finally, our approach was implemented on KAD and on the Gnutella DHT. We successfully tested our solution on KAD against real attacks and we improved its applicability to other DHTs by defining methods to dynamically adapt the detection window to the DHT size of the P2P networks.

Our future work will consist in a better characterization of the different suspicious peers in the DHT (IP addresses involved, number of nodes inserted, prefix used, etc.) in order to clearly identify their number and strategy. Moreover, we will try to find out what are their intents when targeting the DHT by initiating direct communications with them through the P2P protocol. In particular, we are interested in finding out if they are actually monitoring or attacking contents. We will also pursue our crawl and quantification of DHT attacks operations in order to see their evolution in the long run and investigate if the deployment of our protection affects the attackers practices.

References

[CCF09] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. Evaluation of Sybil Attacks Protection Schemes in KAD. In *3rd International Conference on Autonomous Infrastructure, Management and Security - AIMS 2009 Scalability of*

Networks and Services, volume 5637 of *Lecture Notes in Computer Science*, pages 70–82, Enschede Pays-Bas, 2009. University of Twente, Springer.

[CCF10a] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. Efficient DHT attack mitigation through peers' ID distribution. In *Seventh International Workshop on Hot Topics in Peer-to-Peer Systems - HotP2P 2010*, Atlanta USA, 04 2010. IEEE International Parallel & Distributed Processing Symposium.

[CCF10b] Thibault Cholez, Isabelle Chrisment, and Olivier Festor. Monitoring and Controlling Content Access in KAD. In *International Conference on Communications - ICC 2010*, Capetown South Africa, May 2010. IEEE.

[DH06] Jochen Dinger and Hannes Hartenstein. Defending the sybil attack in P2P networks: taxonomy, challenges, and a proposal for self-registration. In *First International Conference on Availability, Reliability and Security (ARES 2006)*, pages 756–763, April 2006.

[DLLKA05] George Danezis, Chris Lesniewski-Laas, M. Frans Kaashoek, and Ross J. Anderson. Sybil-resistant dht routing. In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, *ESORICS*, volume 3679 of *Lecture Notes in Computer Science*, pages 305–318. Springer, 2005.

[Dou02] John 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.

[KLR09] Michael Kohnen, Mike Leske, and Erwin P. Rathgeb. Conducting and optimizing eclipse attacks in the KAD peer-to-peer network. In *NETWORKING '09: Proceedings of the 8th International IFIP-TC 6 Networking Conference*, pages 104–116, Berlin, Heidelberg, 2009. Springer-Verlag.

[LBLLF+10] Stevens Le Blond, Arnaud Legout, Fabrice Le Fessant, Walid Dabbous, and Mohamed Ali Kaafar. Spying the World from your Laptop – Identifying and Profiling Content Providers and Big Downloaders in BitTorrent. In *3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'10)*, San Jose, CA, USA, April 2010. Usenix.

[LMSW10] Thomas Locher, David Mysicka, Stefan Schmid, and Roger Wattenhofer. Poisoning the Kad Network. In *11th International Conference on Distributed Computing and Networking (ICDCN), Kolkata, India*, January 2010.

[LMT08] Francois Lesueur, Ludovic Mé, and Valérie 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, Italy, June 2008. IEEE Computer Society.

[LNR06] Jian Liang, Naoum Naoumov, and Keith W. Ross. The index poisoning attack in p2p file sharing systems. In *INFOCOM*. IEEE Computer Society, IEEE, 2006.

[MM02] Petar Maymounkov and David Mazières. 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.

[MRGS09] Ghulam Memon, Reza Rejaie, Yang Guo, and Daniel Stutzbach. Large-scale monitoring of DHT traffic. In *International Workshop on Peer-to-Peer Systems (IPTPS)*, Boston, MA, April 2009.

[NR06] Naoum Naoumov and Keith Ross. Exploiting p2p systems for ddos attacks. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, page 47, New York, NY, USA, 2006. ACM.

[PKK08] Michael Piatek, Tadayoshi Kohno, and Arvind Krishnamurthy. Challenges and directions for monitoring p2p file sharing networks - or - why my printer received a dmca takedown notice. In *HotSec*. USENIX Association, 2008.

[PSFNR11] Rahul Potharaju, Jeff Seibert, Sonia Fahmy, and Cristina Nita-Rotaru. Omnify: investigating the visibility and effectiveness of copyright monitors. In *Proceedings of the 12th international conference on Passive and active measurement*, PAM'11, pages 122–132, Berlin, Heidelberg, 2011. Springer-Verlag.

[REMP07] Hosam Rowaihy, William Enck, Patrick McDaniel, and Tom La Porta. Limiting sybil attacks in structured p2p networks. In *INFOCOM*, pages 2596–2600. IEEE Computer Society, IEEE, 2007.

[SCDR04] Atul Singh, Miguel Castro, Peter Druschel, and Antony Rowstron. Defending against eclipse attacks on overlay networks. In *EW 11: Proceedings of the 11th workshop on ACM SIGOPS European workshop*, page 21, New York, NY, USA, 2004. ACM.

[SENB07a] Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack. Exploiting kad: possible uses and misuses. *SIGCOMM Comput. Commun. Rev.*, 37(5):65–70, 2007.

[SENB07b] Moritz Steiner, Taoufik En-Najjary, and Ernst W Biersack. A global view of kad. In *IMC 2007, ACM SIGCOMM Internet Measurement Conference, October 23-26, 2007, San Diego, USA*, October 2007.

[SPR09] Georgos Siganos, Josep M. Pujol, and Pablo Rodriguez. Monitoring the bittorrent monitors: A bird's eye view. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 175–184, Berlin, Heidelberg, 2009. Springer-Verlag.

[SR94] R. R. Sokal and F. J. Rohlf. *Biometry: the principles and practice of statistics in biological research.* New York: Freeman, 3rd edition, 1994.

[TCCF11] Juan Pablo Timpanaro, Thibault Cholez, Isabelle Chrisment, and Olivier Festor. When kad meets bittorrent - building a stronger p2p network. In *Eighth International Workshop on Hot Topics in Peer-to-Peer Systems - HotP2P 2011*, Anchorage, USA, April 2011. IEEE International Parallel & Distributed Processing Symposium.

[WTCT+08] Peng Wang, James Tyra, Eric Chan-Tin, Tyson Malchow, Denis Foo Kune, Nicholas Hopper, and Yongdae Kim. Attacking the kad network. In *SecureComm '08: Proceedings of the 4th international conference on Security and privacy in communication netowrks*, pages 1–10, New York, NY, USA, 2008. ACM.

[YFX+09] Jie Yu, Chengfang Fang, Jia Xu, Ee-Chien Chang, and Zhoujun Li. Id repetition in KAD. In Henning Schulzrinne, Karl Aberer, and Anwitaman Datta, editors, *Peer-to-Peer Computing*, pages 111–120. IEEE, 2009.

[YKGF06] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 267–278, New York, NY, USA, 2006. ACM.