Détection de pairs suspects dans le réseau pair à pair KAD

T.Cholez C.Hénard I.Chrisment O.Festor G.Doyen R.Khatoun

présenté à **SAR-SSI 2011**, La Rochelle, France

dans le cadre du projet ACDAP2P (Approche Collaborative pour la Détection d'Attaques dans les réseaux P2P)

du Groupement d'Intérêt Scientifique 3GSS (Surveillance, Sûreté et Sécurité des Grands Systèmes)

19 mai 2011

Nancy-Université Université Henri Poincaré

INRIA

utt université de technologie Troyes

Plan

Plan

Le réseau P2P KAD

KAD est :

- Un réseau pair-à-pair décentralisé (Kademlia DHT)
- Employé pour le partage de fichiers
- Implanté par des clients open source (eMule et aMule)
- Largement utilisé (∼3 à 4 millions de pairs)

Identification des éléments du réseau :

- Chaque élément du réseau (pair, fichier ou mot-clé) possède un identifiant sur 128 bits
- Identifiant des pairs : **KAD ID** (aléatoire)
- Identifiant des fichiers et mots-clés : **hash** (fonction `MD4`)
- Exemple : `MD4`(*avatar*) = `C0F70911A9C2E6F6960DDED0D4118244` en notation hexadécimale
- Zone : subdivision de l'espace d'adressage définie par le premier octet (256 zones, de `00` à `FF`)

Le réseau P2P KAD

Distance logique entre les pairs (métrique XOR) :

- Préfixe : plus grand nombre de bits de poids fort en commun entre deux identifiants
- Exemple : préfixe de longueur 20 (5 caractères hexadécimaux)

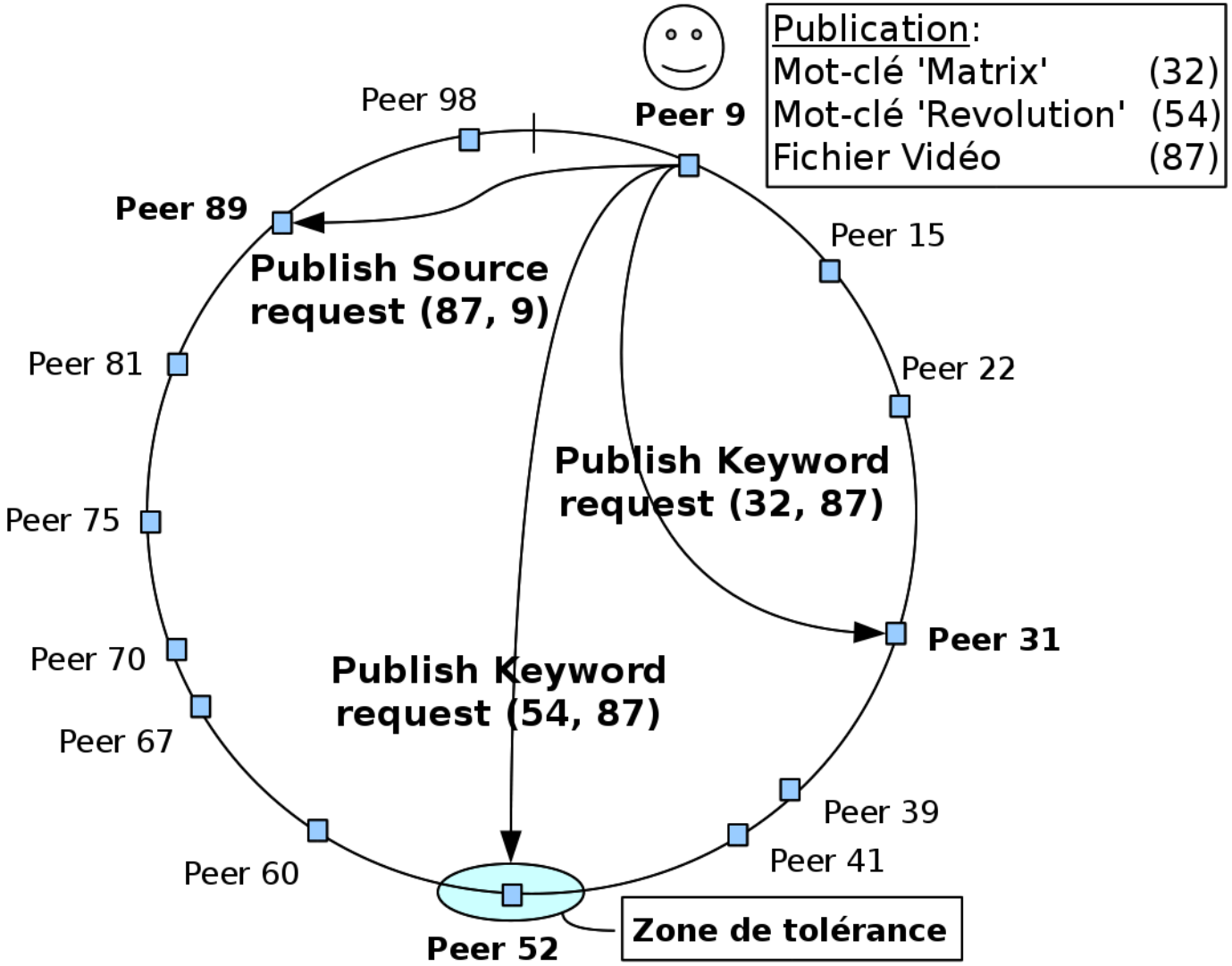
```
C0F70911A9C2E6F6960DDED0D4118244
C0F7073FC6939D0FF3CE0E36B28E3644
```

- Plus le préfixe est grand, plus les éléments sont proches

La DHT de KAD est utilisée pour indexer les contenus :

- Les pairs indexent les ressources du réseau dont les hashs sont proches de leur KAD ID
- Mécanisme à double indexation utilisé pour la recherche et la publication de fichiers (mots-clés → fichiers ; fichiers → sources)

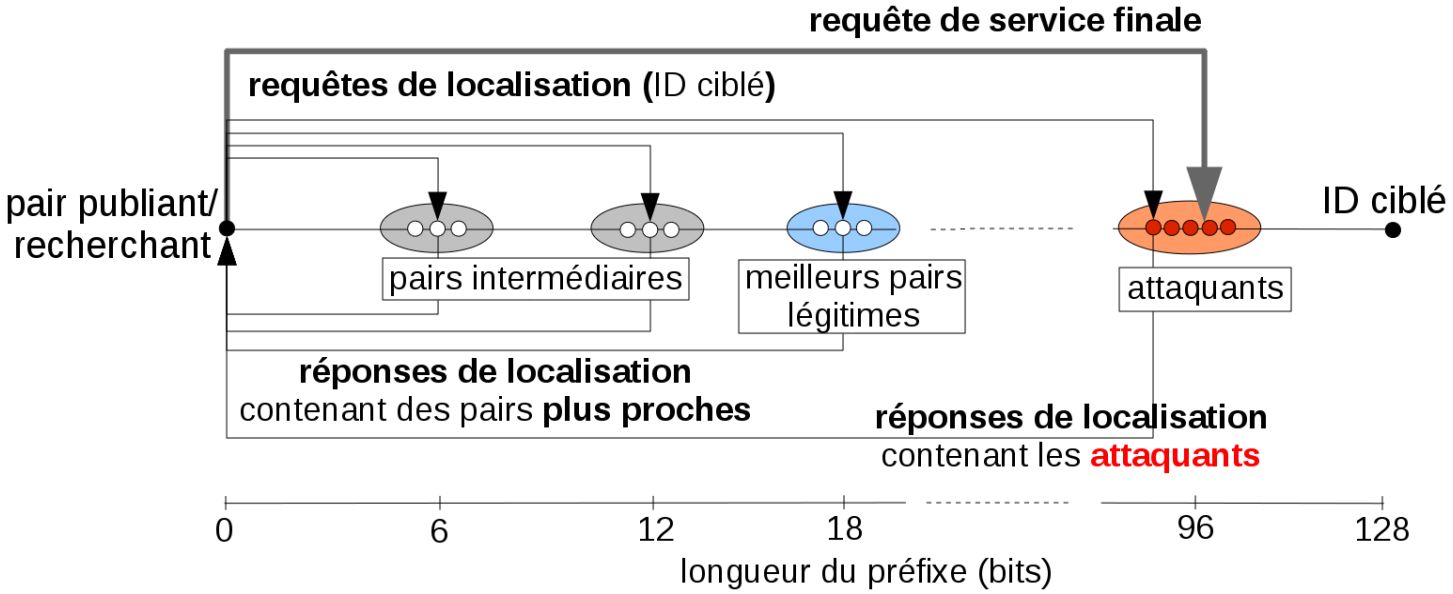
La DHT de KAD


Publication:
Mot-clé 'Matrix' (32)
Mot-clé 'Revolution' (54)
Fichier Vidéo (87)
Peer 9
Peer 98
Peer 89
Publish Source request (87, 9)
Peer 15
Peer 22
Peer 81
Publish Keyword request (32, 87)
Peer 75
Peer 31
Peer 70
Publish Keyword request (54, 87)
Peer 67
Peer 39
Peer 41
Peer 60
Peer 52
Zone de tolérance

Attaques sur KAD

Le réseau KAD est soumis à des attaques :

- Principe :
  - Injection de faux pairs (Sybil attack)
  - À proximité des contenus ciblés
- Applications : espionnage, suppression et pollution de contenu
- Conséquences : détériorations des performances, diffusion de virus, de contenus illégaux et problèmes de vie privée



Attaques sur KAD

Le réseau KAD est soumis à des attaques :

- Principe :
  - Injection de faux pairs (Sybil attack)
  - À proximité des contenus ciblés
- Applications : espionnage, suppression et pollution de contenu
- Conséquences : détériorations des performances, diffusion de virus, de contenus illégaux et problèmes de vie privée

L'État de l'art montre :

- Nombreuses attaques prouvées, aucune réellement recensée
- Nombreuses mesures du réseau, aucune relative à la sécurité

Plan

Qu'est-ce qu'un crawler ?

Un crawler est un outil capable de découvrir l'ensemble des pairs participant à un réseau et de stocker différentes informations les concernant

Objectifs :

1. Collecter des informations sur les pairs participants au réseau KAD (KAD ID, IP, ...)
2. Analyser les données obtenues avec le crawler pour cartographier le réseau et détecter les attaquants ou comportements anormaux

```
<32FFF76959F6A7095347FB338B304330, #.#.#.#, 38060, 16905, 0, T>
<32FFFC5C4D5AE9A082871FF68B1F0D9C, #.#.#.#, 5149, 1025, 4, R>
<32FFFC5C4D5AE9A082871FF68B1F0D9C, #.#.#.#, 5149, 5159, 4, P>
Zone 33: 15196 contacts
<3300048A90460A8AAC3DD2FF542ADF98, #.#.#.#, 12399, 39949, 9, R>
<3300083A0480CFA91B8C142401DD26F2, #.#.#.#, 5611, 5621, 8, T>
<330018506569424D7CBA7133F437EDC8, #.#.#.#, 6647, 6657, 8, P>
<33002596F7AAAA4348FB4349F0A14FA4, #.#.#.#, 46318, 61632, 9, R>
<33002EF905E27753B1900BC602D29C20, #.#.#.#, 19774, 19774, 8, T>
```

Comment explorer le réseau KAD ?

Il existe au sein du protocole de communication de KAD :

- Des requêtes d'amorçage : pour obtenir des informations sur au plus 20 pairs aléatoires
- Des requêtes de routage : pour obtenir des informations sur au plus 20 contacts qui sont les plus proches d'un KAD ID spécifié

$\rightsquigarrow$ utilisées pour découvrir le réseau

La stratégie d'exploration comporte deux phases :

1. Le démarrage : trouver des pairs repartis dans tout le réseau (requêtes de bootstrap)
2. L'exploration complète : obtenir une vue précise de chaque zone du réseau (requêtes Kademlia)

Phase d'amorçage

- Envoi d'une requête de bootstrap à notre client aMule

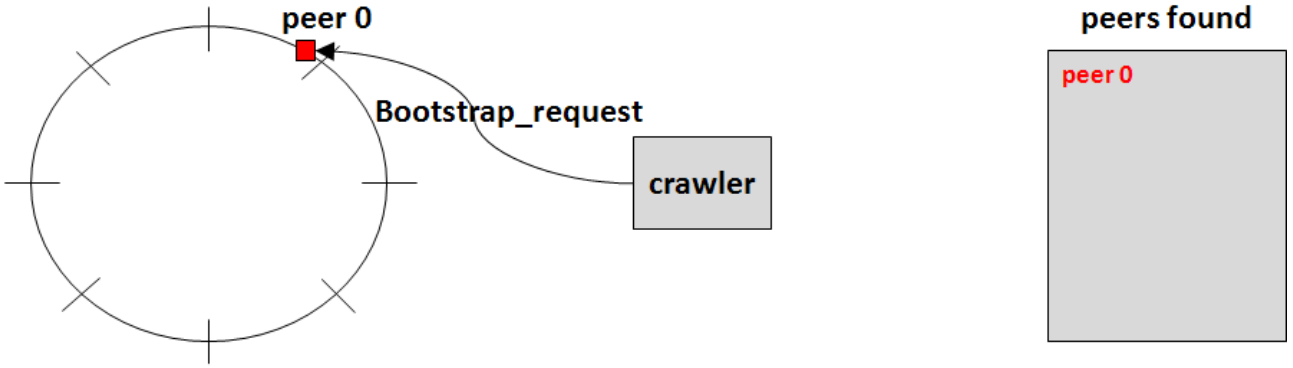


FIGURE: Phase de bootstrap du crawler

Phase d'amorçage

- Envoi d'une requête de bootstrap à notre client aMule
- Réponse : les informations sur 20 pairs, enregistrement des pairs inconnus

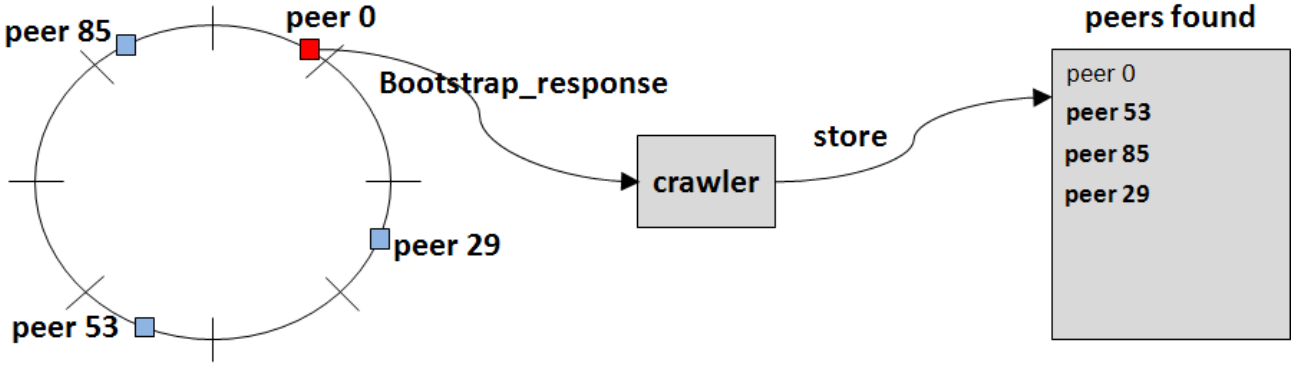


FIGURE: Phase de bootstrap du crawler

Phase d'amorçage

- Envoi d'une requête de bootstrap à notre client aMule
- Réponse : les informations sur 20 pairs, enregistrement des pairs inconnus
- Sélection d'un des pairs trouvé et envoi d'une requête de bootstrap

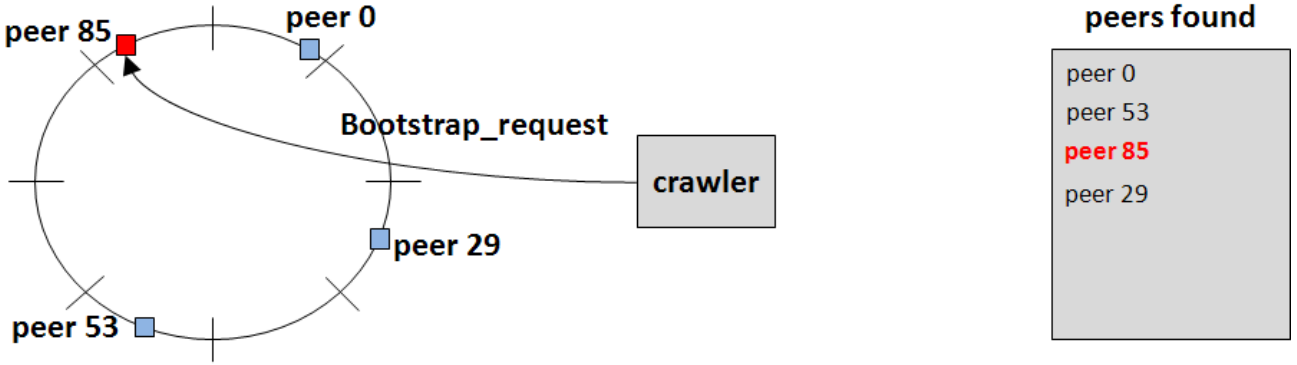


FIGURE: Phase de bootstrap du crawler

Phase d'amorçage

- Envoi d'une requête de bootstrap à notre client aMule
- Réponse : les informations sur 20 pairs, enregistrement des pairs inconnus
- Sélection d'un des pairs trouvé et envoi d'une requête de bootstrap
- Arrêt lorsqu' au moins 500 000 pairs sont trouvés et des pairs dans chaque zone

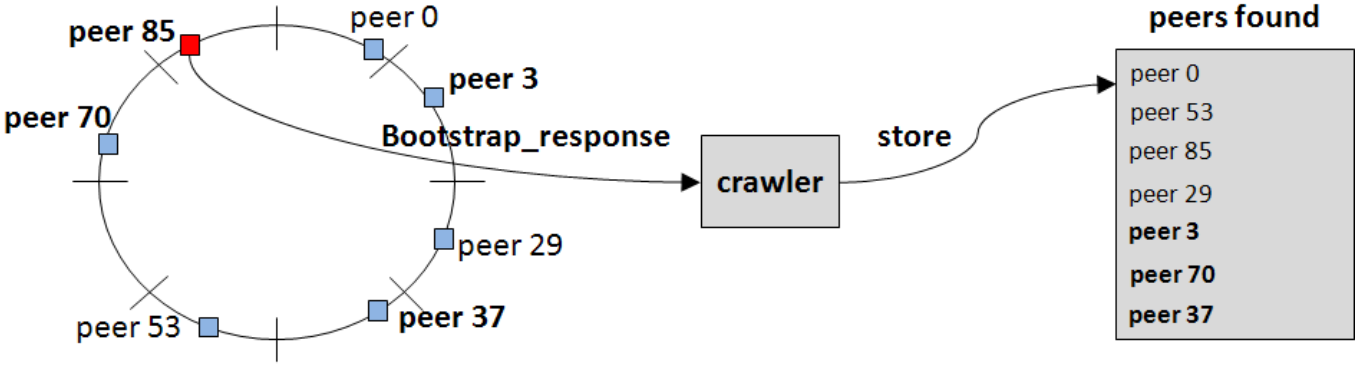


FIGURE: Phase de bootstrap du crawler

Phase d'amorçage

- Envoi d'une requête de bootstrap à notre client aMule
- Réponse : les informations sur 20 pairs, enregistrement des pairs inconnus
- Sélection d'un des pairs trouvé et envoi d'une requête de bootstrap
- Arrêt lorsqu' au moins 500 000 pairs sont trouvés et des pairs dans chaque zone

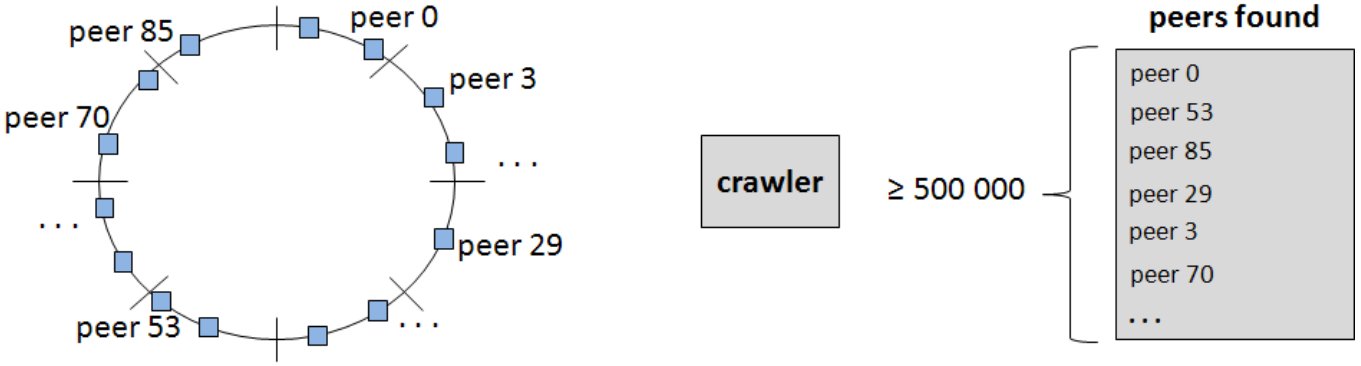
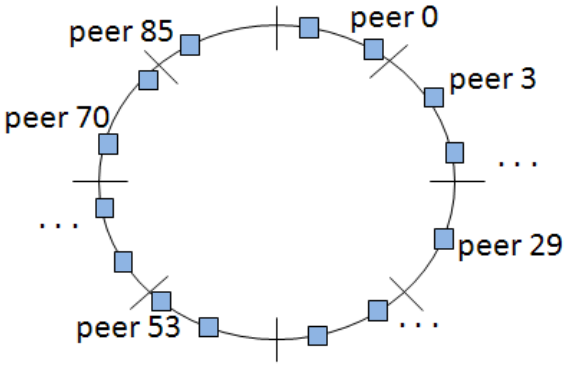
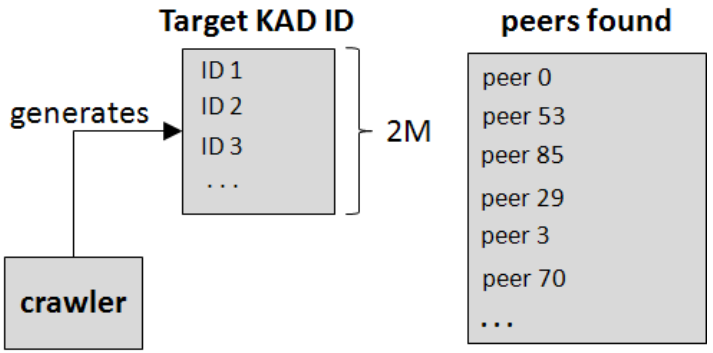


FIGURE: Phase de bootstrap du crawler

Phase d'exploration des zones

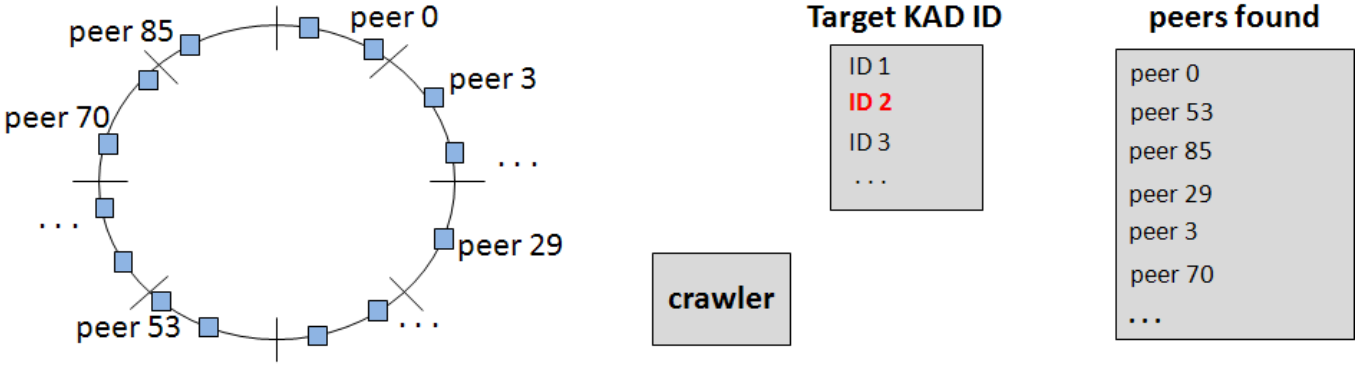
- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :





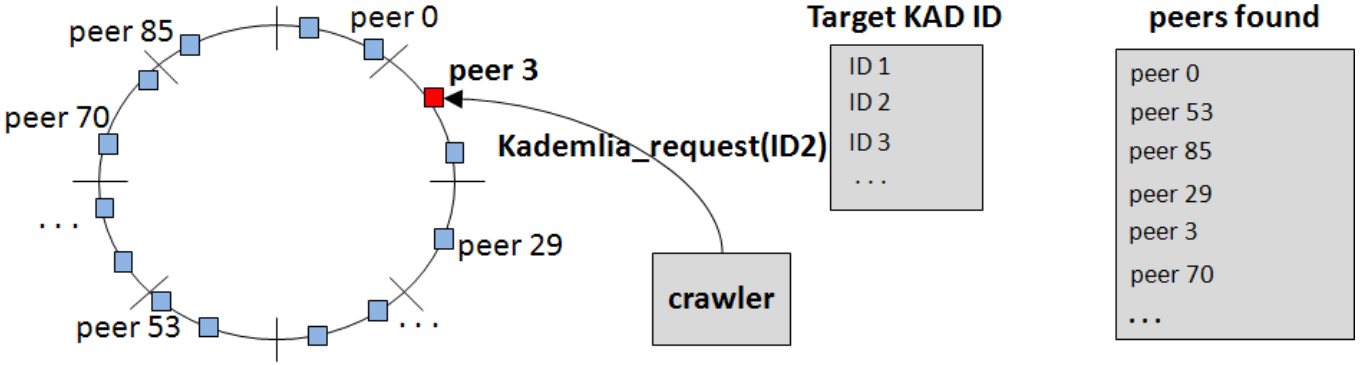
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés



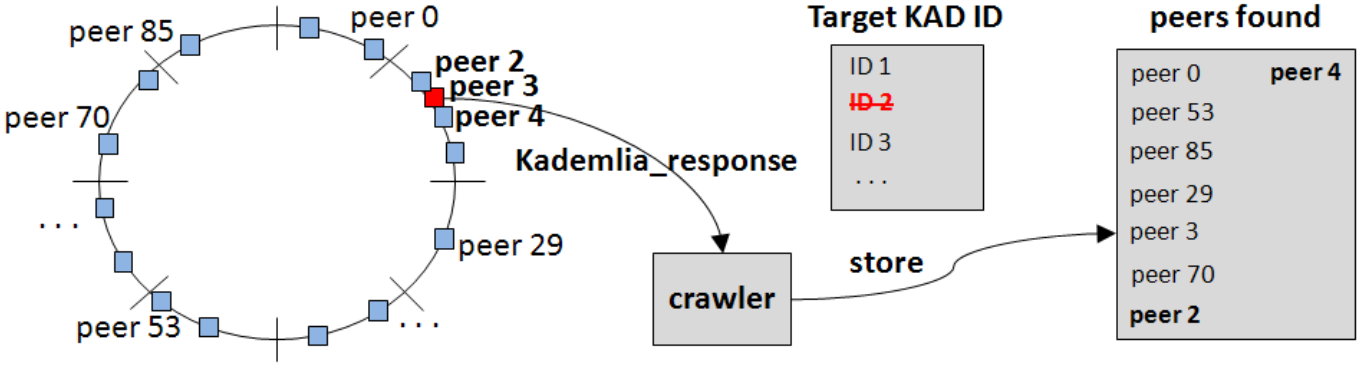
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche



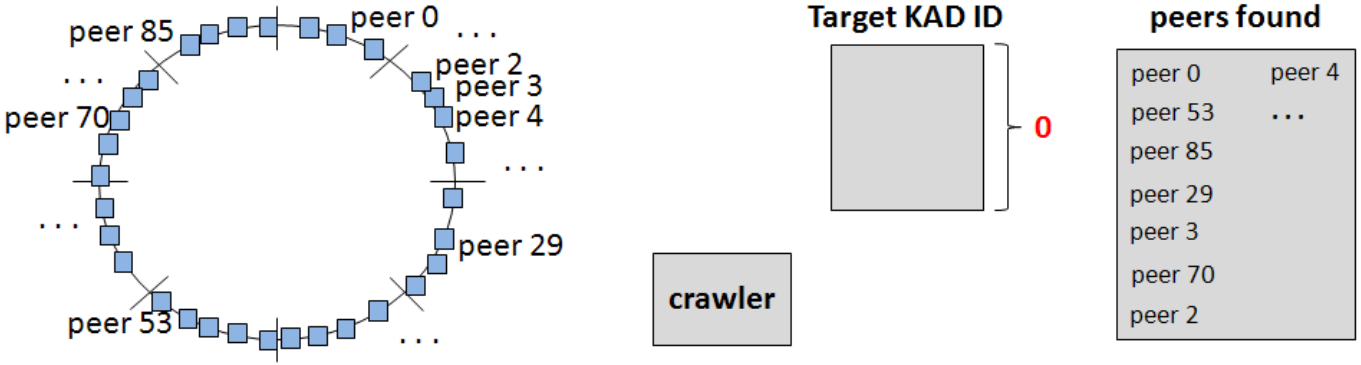
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur



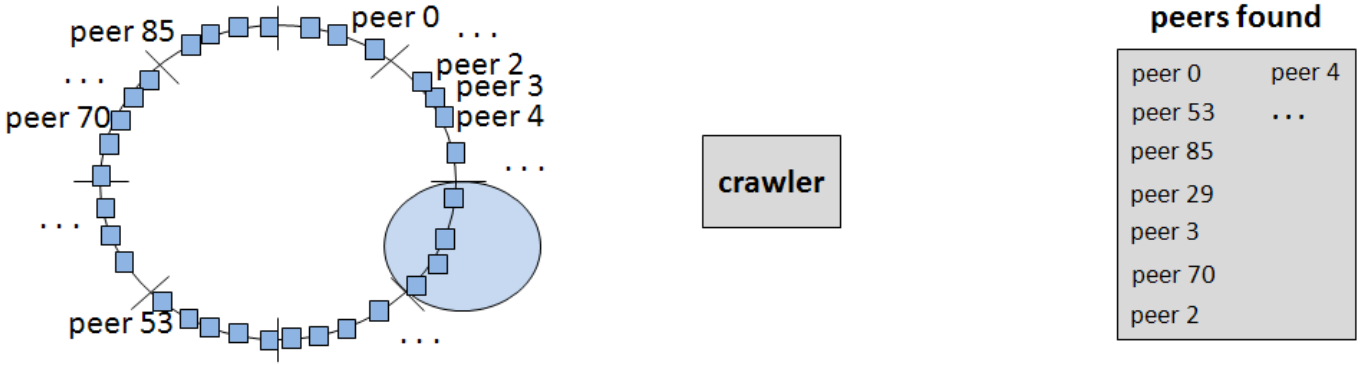
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide



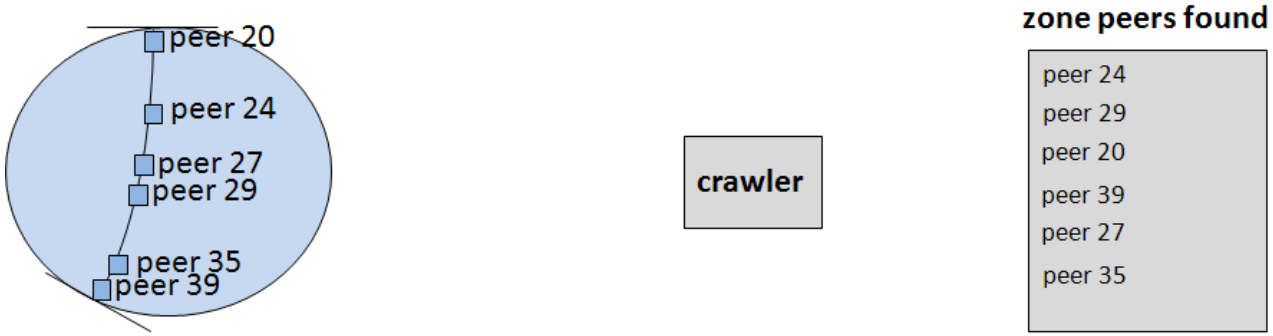
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide



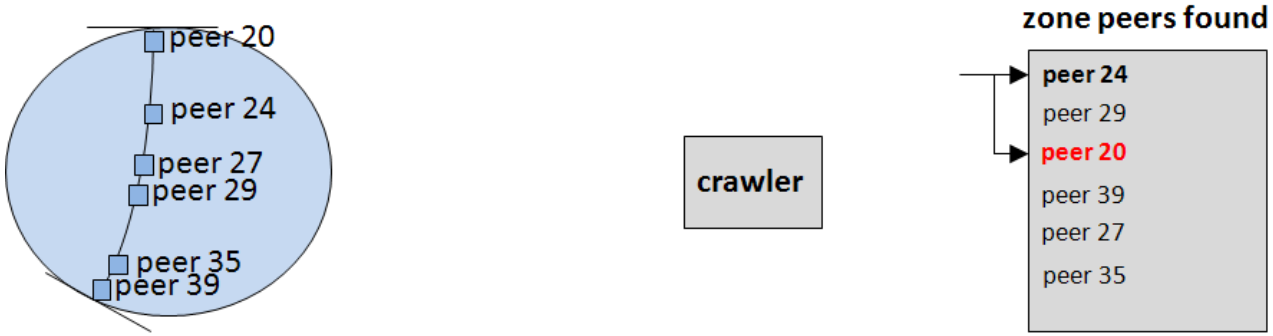
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide
- 2nde passe : optimisation. Pour **chaque pair** trouvé :



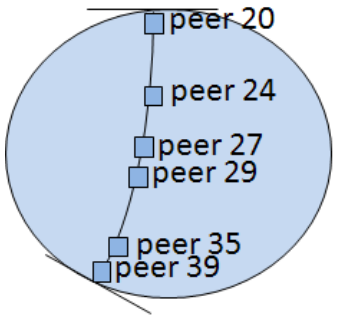
Phase d'exploration des zones

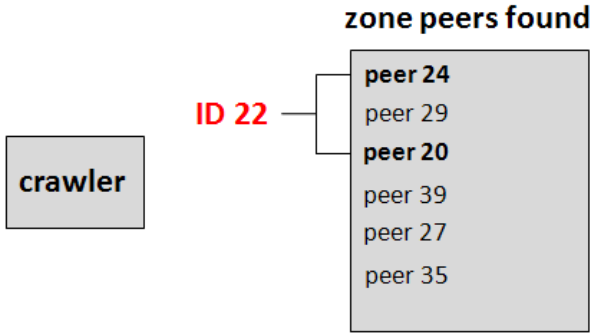
- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide
- 2nde passe : optimisation. Pour **chaque pair** trouvé :
  - Recherche du pair le plus proche



Phase d'exploration des zones

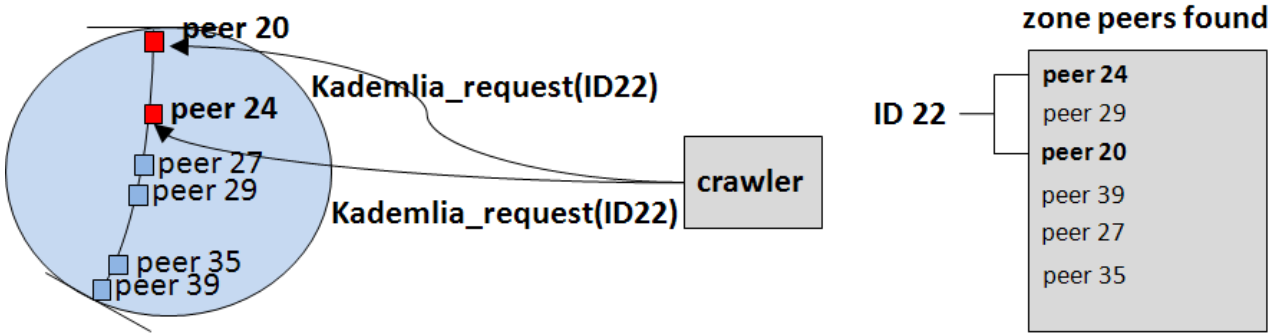
- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide
- 2nde passe : optimisation. Pour **chaque pair** trouvé :
  - Recherche du pair le plus proche
  - Construction d'un KADID entre les deux pairs





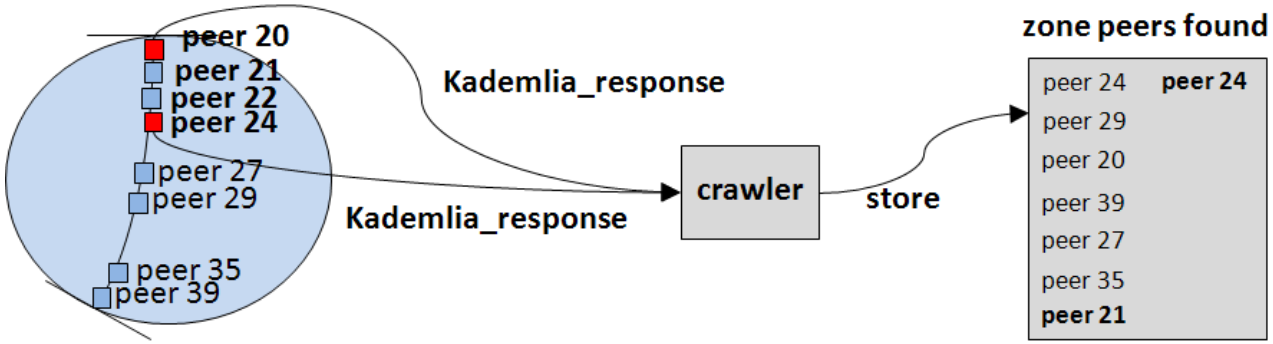
Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide
- 2nde passe : optimisation. Pour **chaque pair** trouvé :
  - Recherche du pair le plus proche
  - Construction d'un KADID entre les deux pairs
  - Envoi d'une requête Kademlia à chacun des deux pairs



Phase d'exploration des zones

- 1ère passe : génèration de ($2^{21} \approx 2M$) de KADIDs cibles bien répartis puis :
  - Sélection d'un KAD ID parmi les pointeurs générés
  - Envoi dans une requête Kademlia au pair le plus proche
  - Si le pair répond, enregistrement des nouveaux contacts et suppression du pointeur, sinon, nouvel essai ultérieur
  - Arrêt lorsque la liste des pointeurs est vide
- 2nde passe : optimisation. Pour **chaque pair** trouvé :
  - Recherche du pair le plus proche
  - Construction d'un KADID entre les deux pairs
  - Envoi d'une requête Kademlia à chacun des deux pairs



Bilan et évaluation

Résultats :

- Entre 13000 et 18000 pairs par zone et entre 3,3M et 4,3M de pairs pour l'ensemble de la DHT
- Pour chaque pair, nous possédons son KAD ID, son IP, les ports TCP, UDP et la version de KAD de son client

Évaluation de la précision du crawler :

- 360 pairs ciblant 72 mots-clés (à 96 bits) ont été déployés (PlanetLab), tous ont été retrouvés
- Instrumentation d'un client normal et comparaison des pairs obtenus

Bilan et évaluation

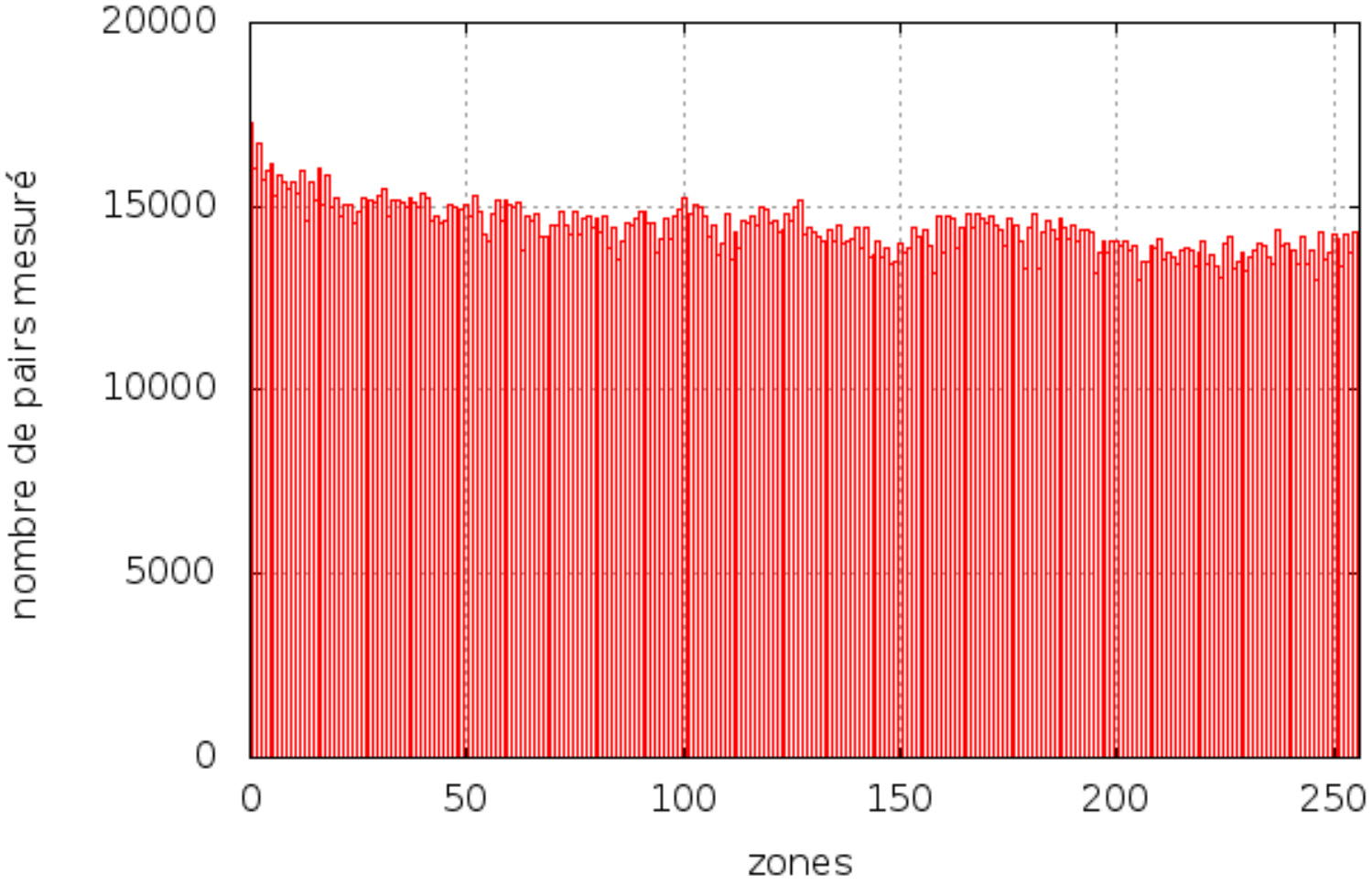
```
[...]
KADID 71: 19856E29730F11CA0E0C210630ADCB36
<19856E29730F11CA0E0C210621142E70, 62.108.171.74, 14337, 13602, 8, T> [99]
<19856E29730F11CA0E0C2106546F8C89, 193.167.187.186, 14690, 13799, 8, T> [97]
<19856E29730F11CA0E0C21065622F60F, 155.245.47.241, 13953, 13779, 8, T> [97]
<19856E29730F11CA0E0C210676E74885, 212.51.218.235, 13897, 14465, 8, T> [97]
<19856E29730F11CA0E0C21069636476A, 129.97.74.14, 14308, 13853, 8, T> [96]
KADID 72: EBCBA6D72037ED01F56809A9FFE6A86E
<EBCBA6D72037ED01F56809A9268DA7FB, 155.245.47.241, 13915, 13842, 8, T> [96]
<EBCBA6D72037ED01F56809A94519B1D4, 129.97.74.14, 14029, 13914, 8, T> [96]
<EBCBA6D72037ED01F56809A9702F72B7, 193.167.187.186, 13666, 14427, 8, T> [96]
<EBCBA6D72037ED01F56809A9892C91A4, 62.108.171.74, 13853, 14683, 8, T> [97]
<EBCBA6D72037ED01F56809A9BAD2A19E, 212.51.218.235, 13861, 13939, 8, R> [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)
21 subnets /24  (showing 362 unique KADIDs in the whole crawler's data)
```

Plan

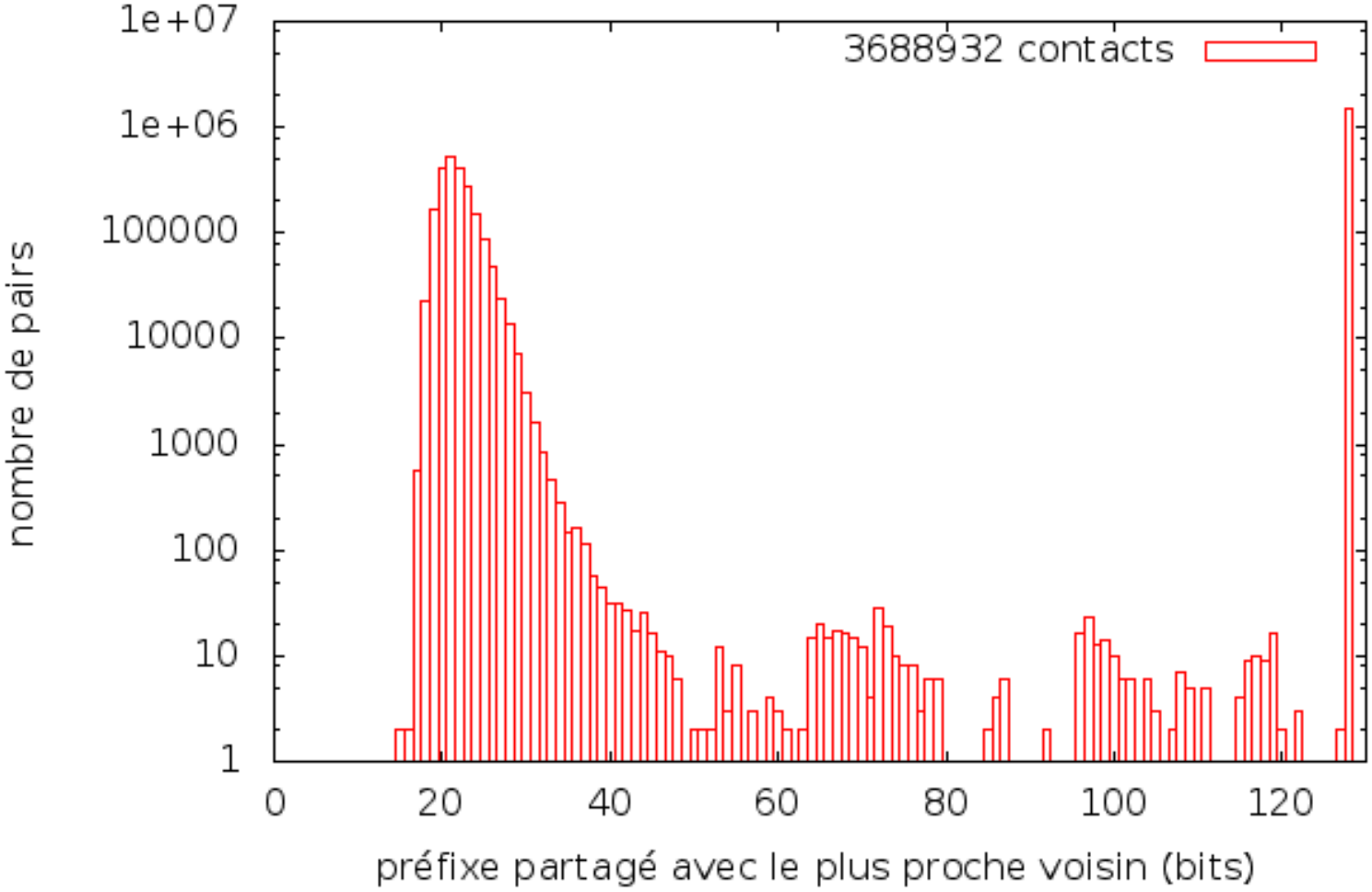
Informations générales

Distribution des 3688932 pairs sur la DHT (le 8 Juillet 2010) :



Analyse des distances entre pairs

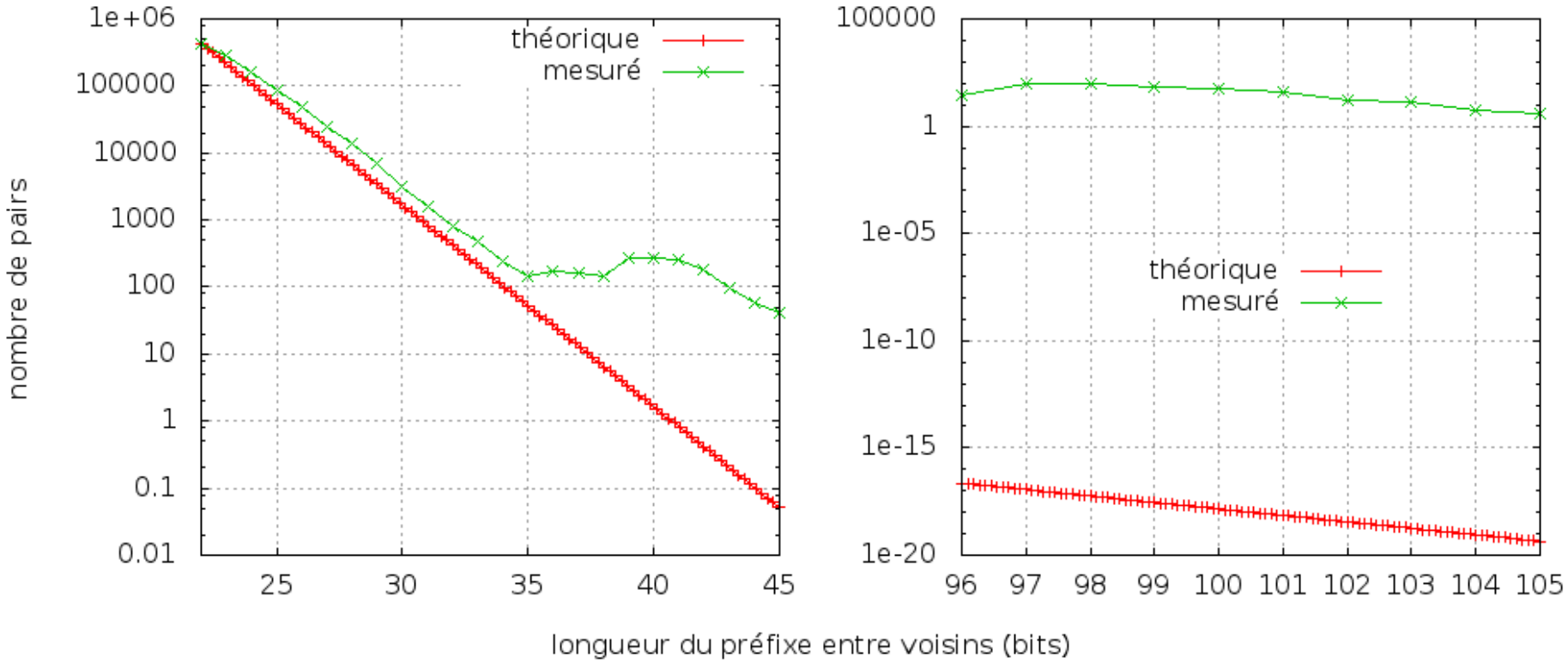
Distribution distances entre voisins la DHT :



Distribution théorique des distances

Nombre moyen de pairs potentiels paratageant x bits avec un identifiant cible étant donné N pairs dans le réseau :

$$F(x) = \frac{N}{2^x} \ avec \ N = 4 \times 10^6 \ et \ x \in [1; 128] \qquad (1)$$



Analyse des distances entre pairs

Étude des groupes de pairs partageant un même (grand) préfixe :

- Au-delà de 35 bits, la longueur d'un préfixe est très suspecte
- Bloc : représente l'ensemble des pairs partageant un préfixe donné
- Permet de trouver les groupes d'attaquants
- Exemple : notre attaque à 96 bits sur le hash du mot-clé *avatar*

```
C0F70911A9C2E6F6960DDED000000000, length 96, shared by 4 contacts:

<C0F70911A9C2E6F6960DDED0213041F2, 193.174.67.186, 34389, 34215, 8>
<C0F70911A9C2E6F6960DDED0B2BCC10E, 152.81.47.4,    33879, 34644, 8>
<C0F70911A9C2E6F6960DDED0D41D0218, 194.167.254.18, 33569, 34263, 8>
<C0F70911A9C2E6F6960DDED0FAD8C991, 142.104.21.245, 34214, 34901, 8>
```

- 426 blocs dont la longueur du préfixe est comprise entre 35 et 127 : autant d'attaques potentielles
- Peu d'attaques évidentes

Évolution des attaques

- Juillet 2010 : 426 groupes de pairs suspects
- Avril 2011 : 2074 groupes de pairs suspects, motifs des attaques évidents

```
Prefix "4A9D8C8777000000000000000000000", length 40, shared by 6 contacts:
<4A9D8C87774AF8C551FE78BDDC3F5A37, 123.144.174.128, 10875, 10875, 8, T>
<4A9D8C877780DFB9985E75EE92AD1C68, 123.144.160.21, 10875, 10875, 8, T>
<4A9D8C877780DFB9985E75EE92AD1C68, 123.145.184.122, 10875, 10875, 8, T>
4A9D8C877797D58D4C21B5BD5224F067, 123.144.160.98, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.167.199, 10875, 10875, 8, T>
<4A9D8C8777F0F03BD1FE123548E269D2, 123.144.163.209, 10839, 10839, 0, R>

Prefix "4A9D8C87778000000000000000000000", length 41, shared by 4 contacts:
<4A9D8C877780DFB9985E75EE92AD1C68, 123.145.184.122, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.160.98, 10875, 10875, 8, T>
<4A9D8C877797D58D4C21B5BD5224F067, 123.144.167.199, 10875, 10875, 8, T>
<4A9D8C8777F0F03BD1FE123548E269D2, 123.144.163.209, 10839, 10839, 0, R>
```

- Limite : au moins 2 pairs insérés pour permettre la détection
- Méconnaissance des contenus ciblés

Analyse des distances entre pairs et contenus

- **Hypothèse** : Les attaquants ciblent les contenus populaires
- Collecte de 888 mots-clés populaires (Amazon, iTunes, PirateBay)
- Recensement des pairs partageant plus de 35 bits avec un contenu

```
twilight 4D62D26BB2A686195DA7078D3720F60A
<4D62D26BB2A686195DA7078D3720F632, X.Y.#.#, 7290, 7294, 8, R> [prefix = 122]
soundtrack AC213377BB53F608390BD94A6AE6DD35
<AC213377BB53F608390BD94A82582F42, #.#.#.#, 5003, 5002, 8, R> [prefix = 96]
harry 770CF5279AB34348C8FECF9672747B94
<770CF5279AB34348C8FECF96524D8CDE, #.#.#.#, 5003, 5002, 8, P> [prefix = 98]
robin B9DF47E5BFAD75F8EE5E3F50EA217983
<B9DF47E5BFAD75F8EE5E3F5051F34AA8, #.#.#.#, 5003, 5002, 8, R> [prefix = 96]
<B9DF47E5BFAD75F8EE5E3F50EA21799F, X.Y.#.#, 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)
41 subnets /24  (showing 2155 unique KADIDs in the whole crawler's data)
```

Analyse des distances entre pairs et contenus

Résultats :

- Un quart des mots-clés attaqués à au moins 96 bits !
- Attaquants présents sur plusieurs mots-clés, motifs identifiables
- 216 mots-clés attaqués = 10% des hashs sur lesquels ces IP sont présentes : potentiellement 2220 contenus attaqués
- Nombreuses attaques pécédemment invisibles (1 seul pair inséré)
- Limites de l'approche : mots-clés chinois, FileID, etc.

mots-clés	préfixe
avatar	126
invictus	123
sherlock	122
princess	122
ncis	96
nero	96

mots-clés	préfixe
nine	122
love	122
american	97
russian	97
black	96
pirate	96

Plan

Résumé et travaux futurs

Résultats :

- Nombreux groupes de pairs suspects (400 en juillet 2010, 2100 en avril 2011)
- Nombreux contenus attaqués (au moins 2200)
- Approches complémentaires, évolution des attaques dans le temps

Travaux futurs :

- Caractérisation des motifs de chaque attaque (IP, ports, distance, etc.), des contenus ciblés
- Commuication avec les pairs suspects :
  - Étude des comportements (surveillance, pollution, DDoS, etc.)
  - Étude des moyens de mise en oeuvre
- Étude à long terme des attaquants (évolutions des contenus ciblés, stratégies d'attaques)
- Développement de mécanismes de protection