

# Lying to the Neighbours

*Nasty effects with tracker-less BitTorrent*

27C3

<astro@spaceboyz.net>

# Disclaimer

## Cyber attacks by WikiLeaks' defenders hit online traders badly

Traders report big drop in sales this week, when attacks on credit card companies started

---

**Graham Snowdon**

[guardian.co.uk](http://guardian.co.uk), Friday 10 December 2010 18.37 GMT

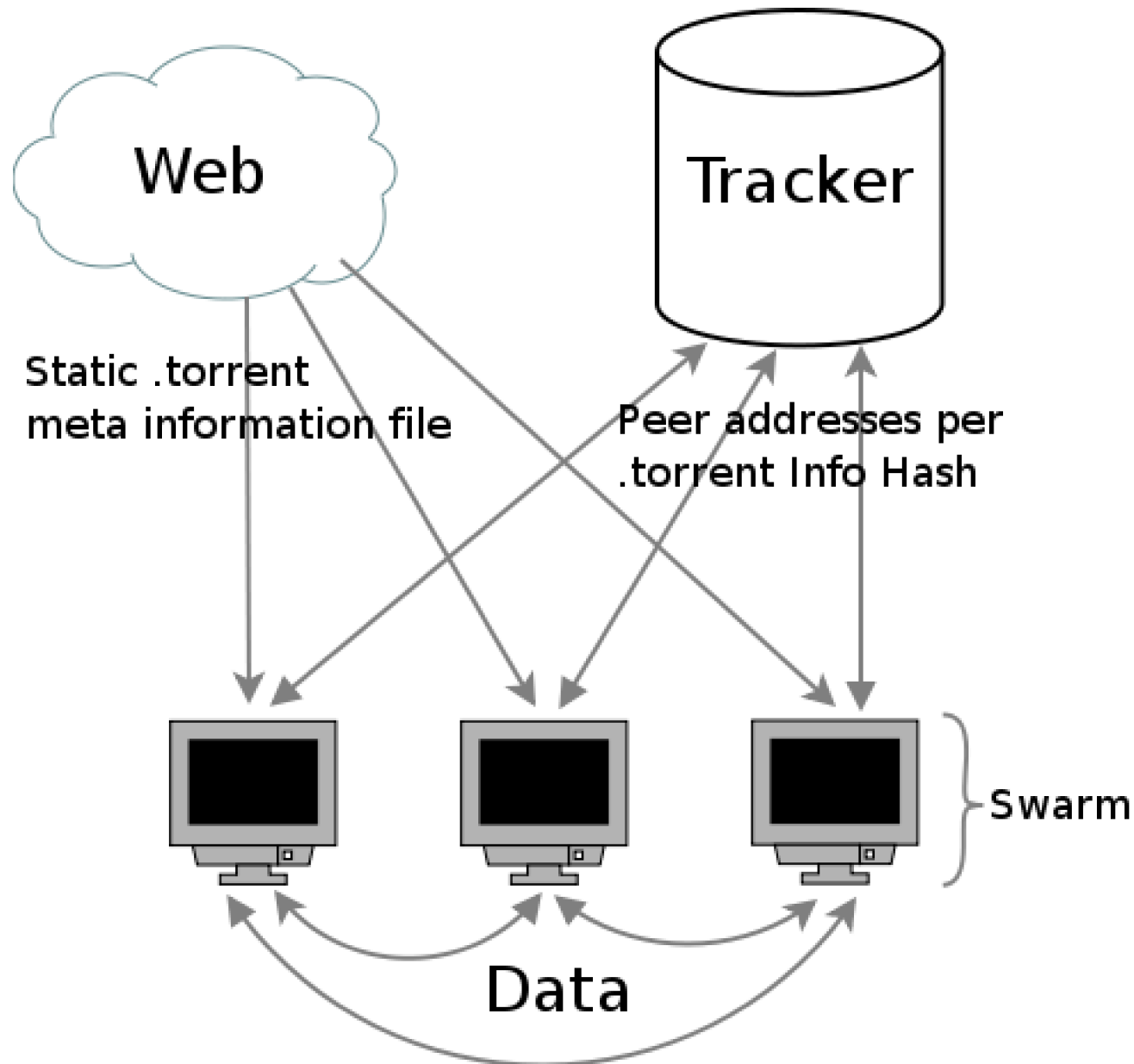
[Article history](#)

---



Emma Louise Ewing, owner of the Kitty Cat Boutique in Falkirk. Her online sales dropped to zero on Wednesday.

# How BitTorrent Works



# .torrent files

- Bencoding:

- `i23e`
- `6:string`
- `litem1...e`
- `dkey1value1...e`

```
d8:announce37:http://tracker.vodo.net:6970/announce7:comment35:vodo / created with libv2v for vodo13:creation datei1292347903e4:infod5:file sld6:lengthi1173790774e4:path l37:Pioneer.One.S01E02.720p.x264-VOD0.mkveed6:lengthi37547969e4:pathl6:Sample44:Pioneer.One.S01E02.720p.x264.Sample-VOD0.mkveed6:lengthi65e4:path l54:Support.Pioneer.One_Make.future.episodes.possible!.URL eed6:lengthi6517e4:pathl13:vodo_720p.nfoeee4:name33:Pioneer.One.S01E02.720p.x264-VOD012:piece lengthi262144e6:pieces92420:
```

# .torrent Info Hash

- ▼ **d** root {4}
  - s** announce (37) = http://tracker.vod
  - s** comment (35) = vodo / created wi
  - n** creation date = 1292347903
  - ▼ **d** info {4}
    - ▼ **l** files [4]
      - ▼ **d** {2}
        - n** length = 1173790774
        - ▼ **l** path [1]
          - s** (37) Pioneer.One.S01E0
      - ▷ **d** {2}
      - ▷ **d** {2}
      - ▷ **d** {2}
    - s** name (33) = Pioneer.One.S01E0
    - n** piece length = 262144
    - s** pieces (92420) = "6bd80d1ed9c

InfoHash =  
SHA1 of  
bencoded  
info value

# Alternative to Indexes

- No central hosting of .torrent
- Link form
- **magnet:**?  
`xt=urn:btih:07a9de9750158471c3302e4e95edb1107f980fa6&dn=Pioneer+One+S01E01+720p+x264+VOD0&tr=http%3A%2F%2Ftracker.publicbt.com%2Fannounce&tr=http%3A%2F%2Ftracker.mytorrenttracker.com%3A6099%2Fannounce`

# Alternatives to Trackers

- Peer Exchange (PEX)
  - Works in the Swarm
  - A client needs  $\geq 1$  Peer
- Distributed Hash Table (DHT)
  - Globally
  - Kademlia Algorithm

# Questions of DHTs

- Who stores data?
  - Everyone
- How to interact with data?
  - By asking one's way

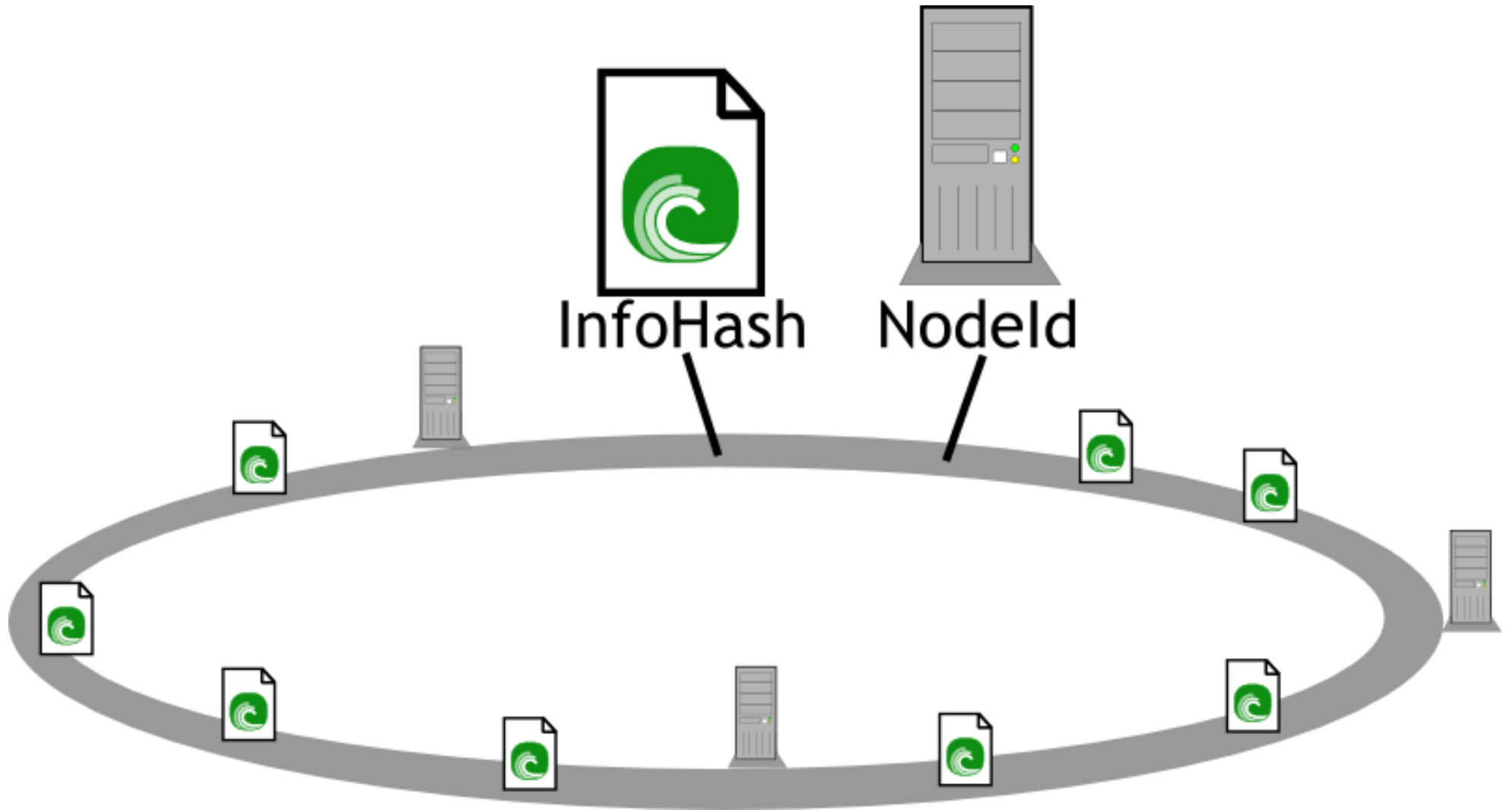
# The BitTorrent DHT

- 160 bits key space
  - maps to InfoHash
- Protocol: bencoding in UDP
- Operations for Topology:
  - ping
  - find\_node
- Operations on Tracker Data:
  - get\_peers
  - announce\_peer

# 160 bit IDs

- SHA1: 160 bits = 20 bytes
- .torrent InfoHash: SHA1 of content description
- DHT NodeID: random
- Client PeerID:
  - client/version information
  - random

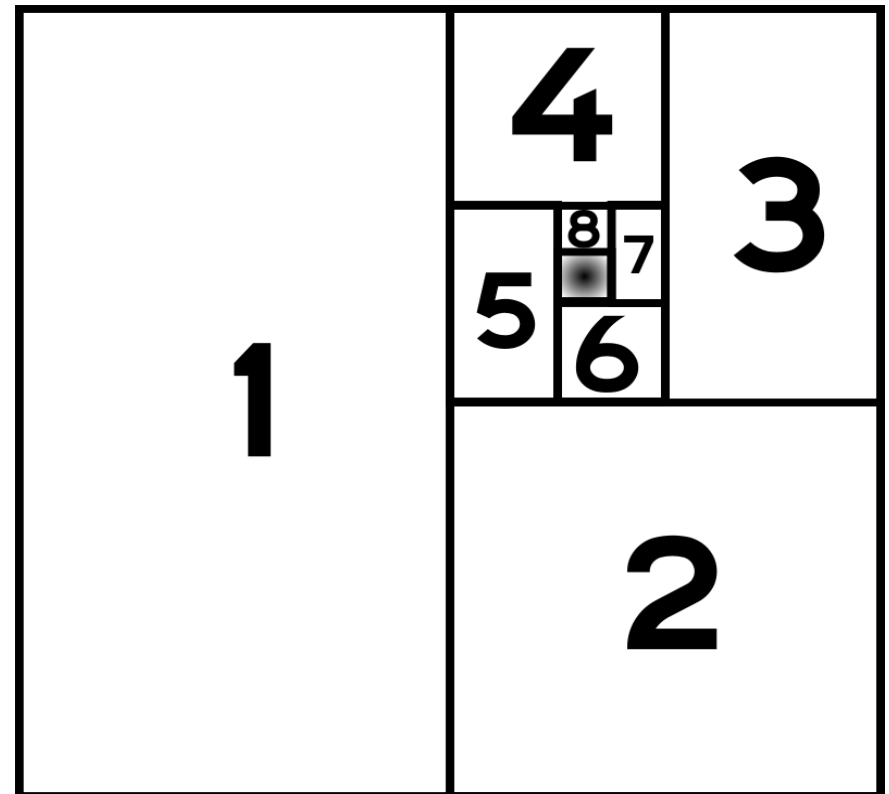
# Key Space



# DHT Routing

- Distance = NodeID  $\oplus$  InfoHash
- Buckets of 8 peers each

Bucket	Distance	Probability
0	1...	1:2
1	01...	1:4
2	001...	1:8
...		
159	00...001	1:1.46*10 <sup>48</sup>



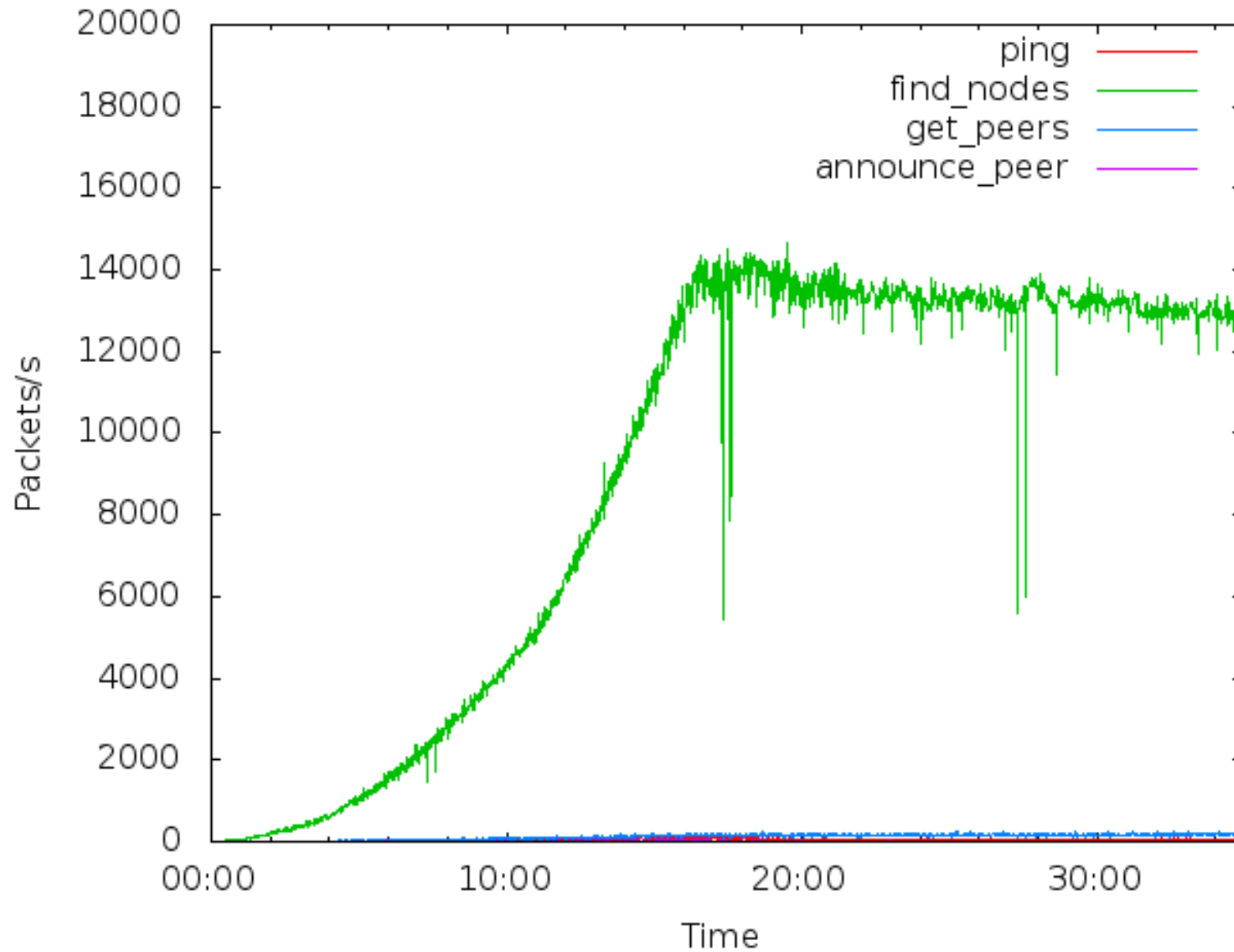
# Topology Requests

- ping
  - To check if a peer is still alive
- find\_node
  - Bootstrapping, “Knitting around holes”, Growing
  - `{"t": "aa", "y": "q", "q": "find_node", "a": {"id": "abcdefghij0123456789", "target": "mnopqrstuvwxyz123456"}}`
  - `{"t": "aa", "y": "r", "r": {"id": "0123456789abcdefghij", "nodes": "def456..."}}`
  - **t**: Request identifier      **y**: Request type      **id**: Sender NodeID
  - **target**: Queried NodeID    **nodes**: 8×(NodeID, IP address, UDP Port)

# Popularity Contest!

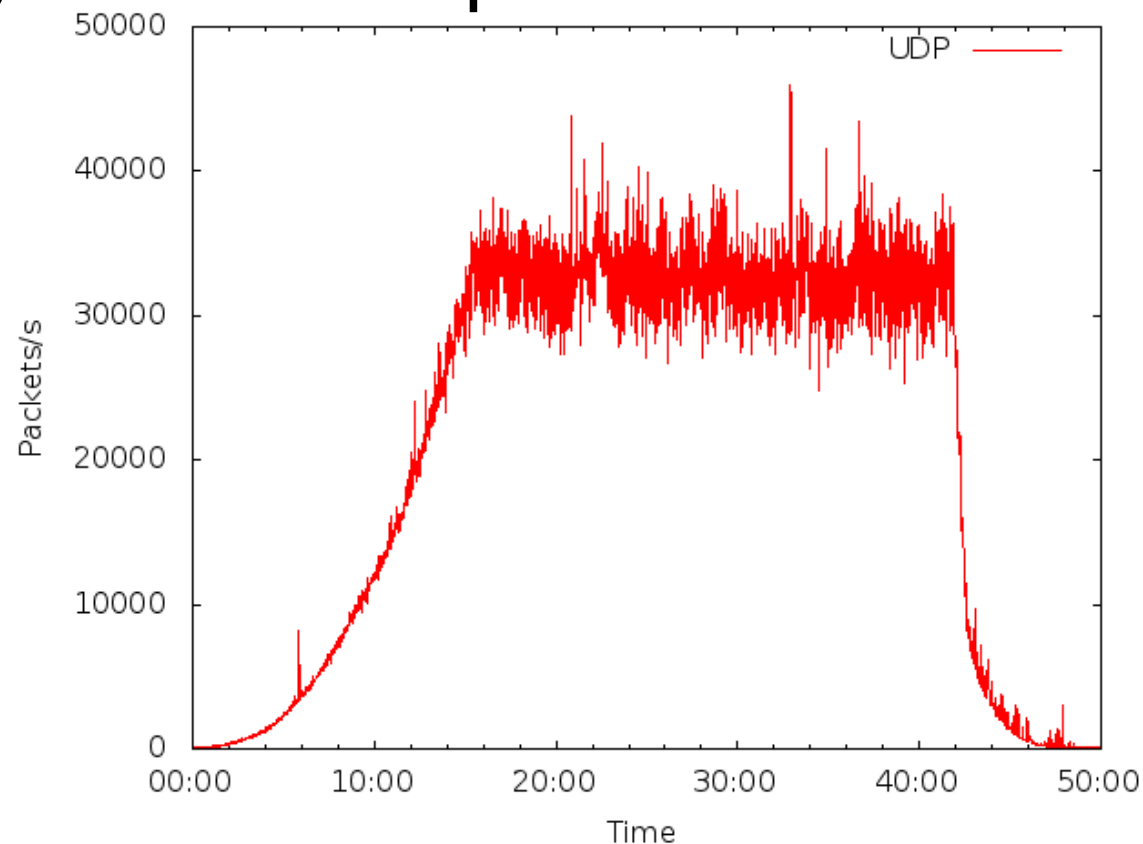
- To facilitate the power of many nodes, we must communicate with many of them.
- *Nearest* peers **least probable**
- Requests & responses contain precious NodeIDs
- Flipping some of the least significant bits results in a immediately neighbouring NodeID

# Being Popular



# Using Popularity

- A neighbouring peer gets *find\_node* requests
- We can put 8 neighbouring NodeIDs and arbitrary addresses/ports into *nodes*



# TCP > UDP











- BitTorrent is about TCP connections
- Spamming trackers with arbitrary IP addresses has been done before
- Can we do the same with tracker-less BitTorrent?

# Sock Puppets!

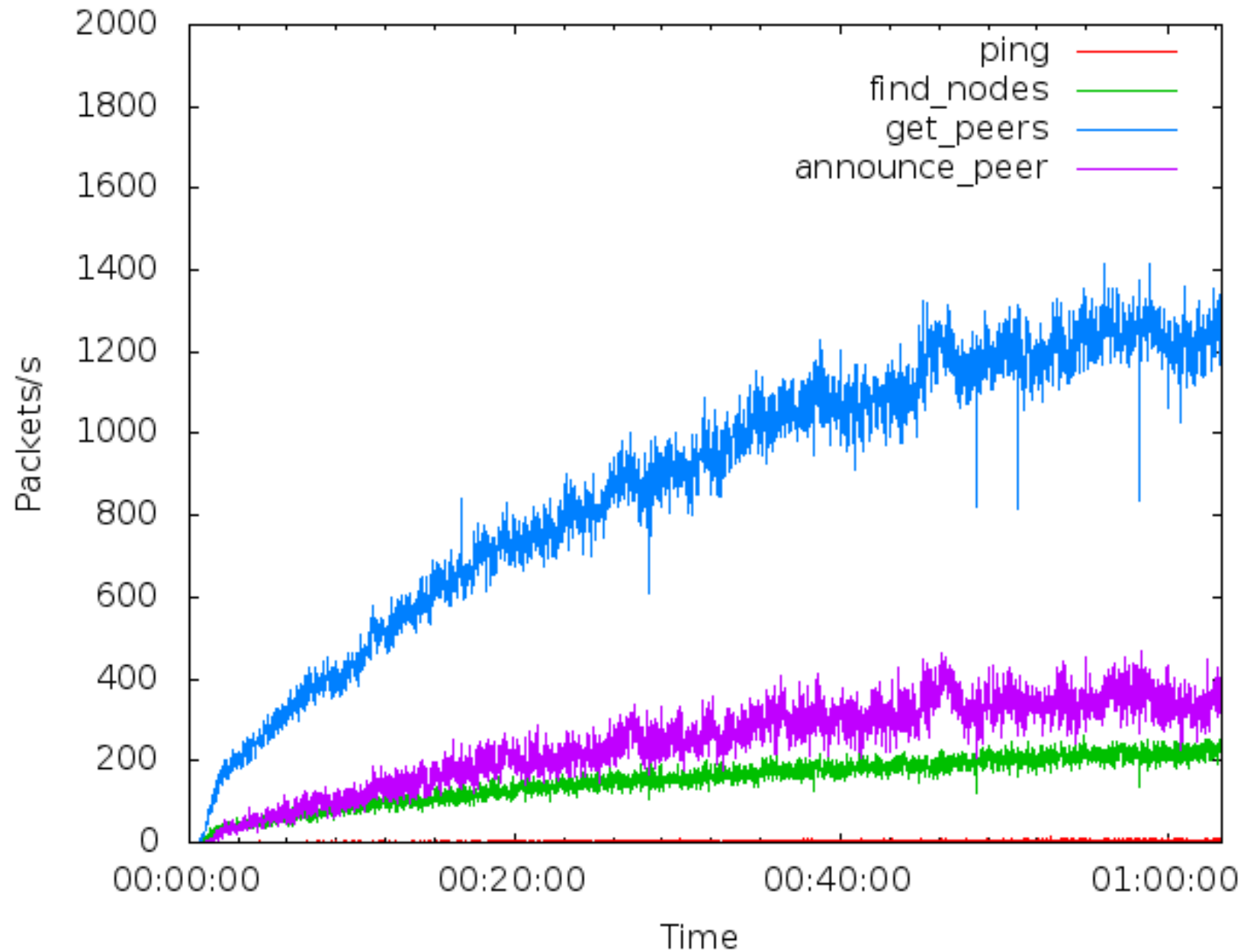


# “Sybil Attack”

- We can take any (“random”) NodeID in the key space
- Nodes serve Tracker data neighbouring their NodeID
- Tracker Data → TCP connections
- Some Info Hashes are quite popular

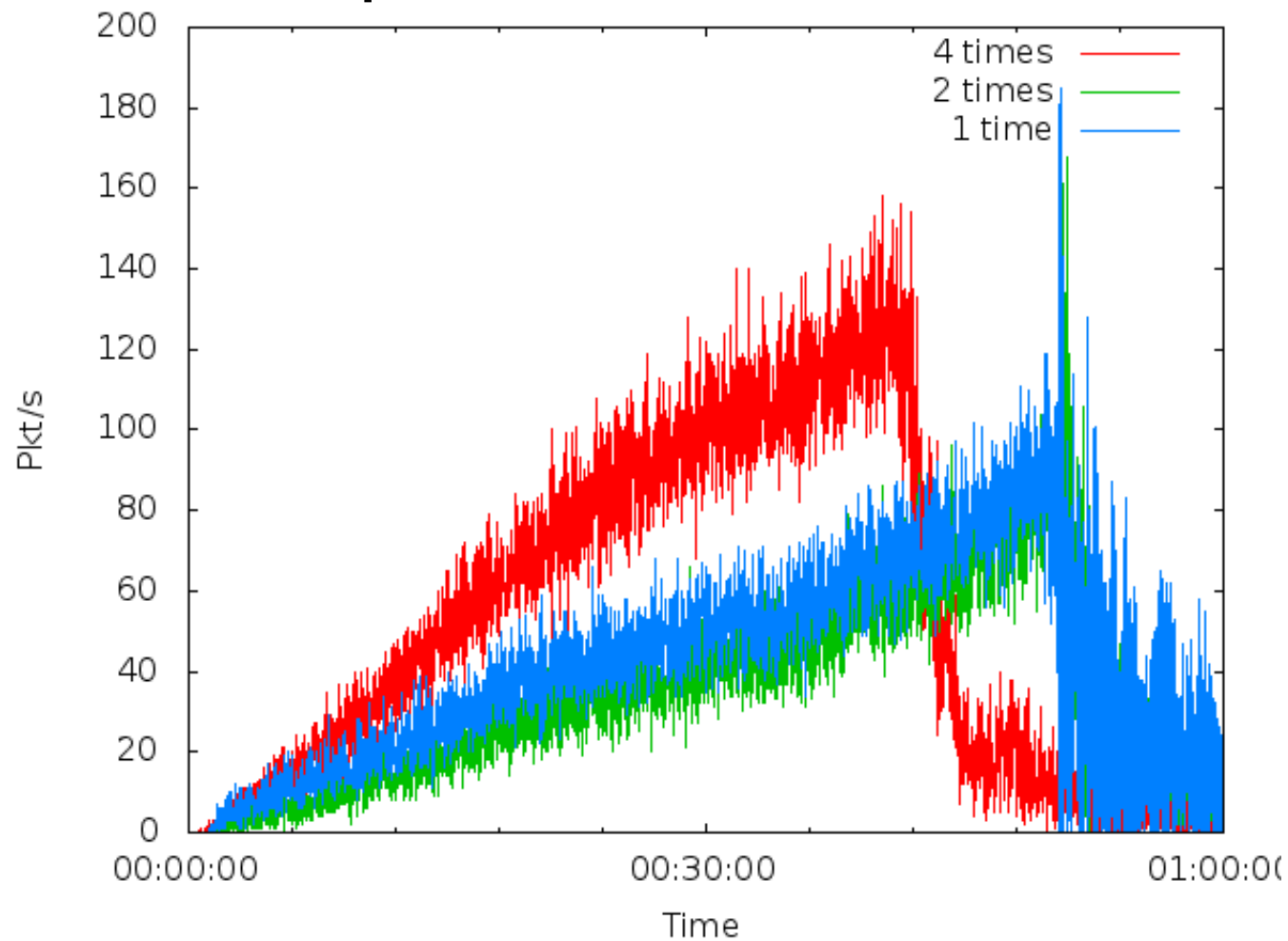
Uploaded		Size	SE	LE
10-30 14:29		1.44 GiB	19040	8104
12-17 03:00		174.71 MiB	13520	886
11-26 07:36		698.73 MiB	11703	1786
12-09 21:13		699.94 MiB	8621	2114
10-27 19:53		713.93 MiB	8302	2784
12-13 04:51		550.16 MiB	8099	666
12-10 03:15			8082	450
12-04 12:18		1.24 GiB	7726	4597
08-28 17:19		1.37 GiB	7462	878
11-21 20:08		716.05 MiB	6903	3568

# 267 most-leeched .torrents on TPB



# Impact

- Many implementations with different caching behaviour & duplicates detection



# Don't Panic

- Being part of a DDoS attack may not be noticeable for a single user with a sane implementation
- DHT is still usable, Trackers are often not
- Client implementors should:
  - Index routing information not only by distance but IP address
  - Avoid communicating with ports  $< 1024$

# A Way Out?

- Another Kademlia user: *telehash.org*
- NodeID = SHA1(address:port)
- Easily verifiable
- Reduces possible NodeIDs per IP address from  $2^{160}$  to  $2^{16}$
- [http://www.rasterbar.com/products/libtorrent/dht\\_sec.html](http://www.rasterbar.com/products/libtorrent/dht_sec.html)