GNUnet Distributed Data Storage
DHT and Distance Vector Transport

Nathan S. Evans\(^1\)

\(^1\)Technische Universität München
Department of Computer Science
Network Architectures and Services

July, 24 2010
Overview

- Distributed Hash Tables
  - Usage
  - Background
  - Kademlia

- Restricted Route Networks

- GNUnet DHT
  - Motivation
  - Design
  - Implementation
  - Issues

- Distance Vector Transport
  - Purpose
  - Design
  - Benefits

- Summary and Conclusion
DHT Basics

- Shared data storage for P2P networks
- Reliably store files, data
- Pool bandwidth, storage space
- Data remains when publisher disappears
- Decentralization
DHT Basics

- Data identified by keys
- Peers store data based on key
- Distributed Key/Value mapping
- Load balancing
- Simple *PUT* / *GET* abstraction
- Scalable
DHT Example - Data XYZ Stored in Network
DHT Example - Data Available when Nodes Lost
DHT Routing

- Routing is the essential internal function of DHT’s
- Enables storage, lookup
- DHT’s have distance metric $D_{a,b}$ where $a, b$ are keys or node identifiers
- Common DHT’s route requests to next closest node w.r.t key

⇒ Provided closest node is found, a **PUT** and subsequent **GET** request for a key will reach the same node, and data will be stored/found properly.
DHT Routing Example - Base Topology
DHT Routing Example - Peer (1, 1, 2, 2)
DHT Routing Example - Data Search 1
DHT Routing Example - Data Search 2
DHT Routing Example - Data Search 3
DHT Routing Example - Data Search 4
DHT Routing Example - Data Search 5
DHT Routing Example - Data Search 6
DHT Routing Example - Data Search 7

![DHT Routing Diagram](image)
DHT Routing Example - Data Search 8
DHT Routing Example - Data Search 9
DHT Routing Example - Data Search 10
DHT Routing Example - Data Search
DHT Limitations

- Not secure
  - Eclipse attacks
  - Block access to data
  - DoS
- Not scalable/decentralized
  - Routing tables and maintenance
  - Require trusted authority
- Require well connected graph
DHT Attack Example - Base Topology
DHT Attack Example - Put Data Normal
DHT Attack Example - Get Data Normal
DHT Attack Example - Random Mallory 1
DHT Attack Example - Random Mallory 2
DHT Attack Example - Random Mallory 3
DHT Attack Example - Sybil and Mallory
DHT Attack Example - Sybil and Mallory 2
DHT Attack Example - Sybil and Mallory 3
DHT Attack Example - Mallory Surround
DHT Attack Example - Mallory Surround 2
DHT Attack Example - Mallory Surround 3
Kademlia Routing

- Routes to peer with most matching bits
- Assume sufficiently full routing tables
- Number of possible nodes is halved at each step
- Results in $O(\log n)$ steps
- Same method used to search a binary tree
- Metric is simple, other DHT’s require multiple routing tables to achieve same performance
Kademlia Peer 101 Routing Table
Kademlia Peer 101 Routing Table
Kademlia Peer 101 Routing Table
Kademlia Peer 101 Routing Table
Kademlia Peer 101 Searching for 010
Kademlia Peer 101 Searching for 010
Kademlia Peer 101 Searching for 010
Kademlia Peer 101 Searching for 010
Kademlia Peer 101 Searching for 010
Kademlia Today

- Kademlia most widely used DHT
- “Kad” network used by eDonkey, eMule, aMule
- DHT used for searching for and storing “links”
- Millions(!) of users
- Problems
  - Highly deterministic routing
  - DoS attacks possible
  - Eclipse, content blocking possible
  - NAT’d users need “buddy” to participate
  - Requires fully connected topology
Restricted Route Network Topologies

- Assumption that any two peers with an IP address can communicate doesn’t always hold
  - NAT
  - Firewalls
  - Mobile Networks
  - Ad-hoc Networks
  - Sensor Networks
- DHT’s commonly assume/require universal connectivity
- We would like our DHT to function in all the above networks
- Build in support for uPnP, NAT punching, etc.
- Goal: provide more connectivity to sparsely connected graphs (DV)
Motivation

- Efficient
- Decentralized
- Scalable
- Secure
  - Reliable
  - Fault-tolerant
  - Provides availability
- Operates in diverse topologies

⇒ Build off previous DHT designs
Kademlia Comparison

■ Kademlia similarities
  ■ Use $\oplus$ distance metric
  ■ Routing table made up of $k$-buckets
    $\Rightarrow$ Bucket $k_i$ holds peers with matching prefix length $i$
  ■ Efficient, binary search like routing ($O(\log n)$)

■ Departure from Kademlia
  ■ Randomized, recursive routing
  ■ Forward requests to multiple peers
  ■ Closest *local peers* to key store data
  ■ Designed to work in diverse networks, even those without universal connectivity
  ■ Routing works with sparse set of connections
GNUnet DHT - Base Example
GNUnet DHT Examples

GNUnet DHT - Insert Example 1 (Normal)

0 1 2 3 4 5
0
1
2
3
4
5

N. Evans (TUM)

GNUnet Distributed Data Storage

July, 24 2010 52 / 102
GNUnet DHT - Insert Example 2 (Randomized)
GNUnet DHT - Insert Example 3 (Randomized)
GNUnet DHT - Insert Example 4 (Randomized)
GNUnet DHT - Insert Example 5 (Randomized)
GNUnet DHT - Insert Example 6 (Failure)
GNUnet DHT - Insert Example 7 (Split)
GNUnet DHT - Insert Example 8 (Randomized and split)
GNUnet DHT - Get Example - Base
GNUnet DHT - Get Example - 1 Malicious
GNUnet DHT - Get Example - Split-route
GNUnet DHT - Get Example - Out-route
GNUnet DHT - Get Example - 3 Malicious
GNUnet DHT - Get Example - Route around
**GNUnet DHT - Surround Data**
GNUnet DHT - Insert Redundancy
GNUnet DHT - Mallory and Sybil fail
Implementation

- DHT implemented in both 0.8 and 0.9 versions of GNUnet
- Kademlia style routing table, randomized recursive routing
- Routing table from connected peers
- Testing and evaluation framework
  - Start GNUnet peers
  - Connect in certain topology
  - Insert/retrieve data, log high and low level results
Remaining Issues

- Determining topology (and therefore network size) is difficult
- Need to know for appropriate routing
- More testing with malicious peers and churn needed
- Routes through network are found, not always efficiently
- Simulate large number of well connected peers

⇒ More complete routing tables ≈ better routing: can we improve our tables?
Fisheye Bounded Distance Vector

- Efficient routes between nodes can be determined using distance vector protocol
  - Every node publishes complete routing information
  - Data disseminated to all nodes; so best routes are found
- Distance vector requires lots of communication and state
- Limit routing information shared by hops to target
- Gives each node a complete “Fisheye” view of their local neighborhood
- Using FBDV in GNUnet provides more peers than available by only direct connections
- Restricted route networks can become better connected!
Design

- Distance vector implemented as transport plugin
- When peers connect, they gossip their known neighbors
- Gossiped peers can be communicated with via gossiper
- Assume a network of three peers, A, B and C
- A connects to B, then B connects to C
Example - Peers A, B, C connected in line
Example - Peer B gossips C to A, A to C
Example - Peers A and C “connected” via B
Example - Peer D joins, connects to C
Example - Peer C gossips B, A to D
Example - D “connected” to B, A via C
Under the Hood

Figure: Services for peers A, B, C
Under the Hood

Figure: CORE A receives message for C
Under the Hood

Figure: CORE A hands to TRANSPORT A
Under the Hood

Figure: **TRANSPORT** A hands to **DV** A
Under the Hood

**Figure:** DV A hands to CORE A
Under the Hood

Figure: **CORE** A hands to **TRANSPORT** A
Under the Hood

Figure: TRANSPORT A hands to TCP A
Under the Hood

Figure: Message sent via TCP to Peer B
Under the Hood

Figure: TCP B hands to TRANSPORT B
Under the Hood

Figure: TRANSPORT B hands to CORE B
Under the Hood

Figure: **CORE B** hands to **DV B**
Under the Hood

Figure: DV B hands to CORE B
Under the Hood

Figure: **CORE B hands to TRANSPORT B**
Under the Hood

**Figure:** TRANSPORT B hands to TCP B
Under the Hood

Figure: Message is sent via TCP to peer C
Under the Hood

Figure: TCP C hands to TRANSPORT C
Under the Hood

Figure: TRANSPORT C hands to CORE C
Under the Hood

Figure: **CORE C hands to DV C**
Under the Hood

Figure: DV C hands to TRANSPORT C
Under the Hood

Figure: TRANSPORT C hands to CORE C
Figure: **CORE C** delivers message from **A**
Benefits of FBDV

- Provide more available peers to DHT
- Local routing very efficient, transport selection further optimizes
- Sparse topologies have better chance for routing
- Allows DHT routing algorithm to remain more general
- Shifts connection overhead from file handles to memory/CPU
- DV messages essentially onion encrypted/routed

⇒ DV may be used in the future for long lived Tor style “circuits” to provide anonymity!
• GNUnet should have a DHT
  • Distributing storage
  • Replication
  • Reliability, scalability

• DHT should be efficient, secure
  • Must compromise
  • Decide which goals most important
  • Build on existing designs
  • Randomized routing

• DHT should work in diverse topologies
  • Requires flexible routing
  • Use DV to help “secluded” peers
Questions?