Anonymity in Structured Peer-to-Peer Networks

Nikita Borisov and Jason Waddle
Introduction

• Existing P2P systems offer anonymity or structured routing, but not both
• We aim to investigate the interaction of network structure and anonymity
• Analyze Chord and some modifications
  – Techniques generalize to other systems
• Focus on source-anonymous DHT lookup
Quantifying Anonymity

• Attacker
  – gets some information about an event
  – wants to learn identity of initiator of that event

• Crowd
  – a set $\Omega$ of potential initiators

• Initiator
  – a member of $\Omega$
  – wants to blend in – not differentiate herself in the eyes of the attacker from the rest of $\Omega$
Quantifying Anonymity

We want to measure the *uncertainty* of the attacker with respect to the identity of the initiator.
Quantifying Anonymity

A
Quantifying Anonymity

A → 01101...
Anonymity Sets

- **Anonymity Set** \( S \) is a random variable
  - the set of initiators that cannot be ruled out as the true initiator based on the information available to the attacker

- Anonymity can be measured as the expected size of this set \( E[|S|] \)
  - Normalized: Anonymity = \( E[|S|]/|\Omega| \)
Anonymity Sets

A \leftarrow 01101\ldots

Anonymity = 9/16
Anonymity Sets

- Anonymity sets do not generalize to a probability distribution $P$ on initiators implied by information available to attacker.
  - e.g., there is an $x \in \Omega$, $P(x) = .99999$
    - $P(\Omega - \{x\})$ uniform (and nonzero)
    - $\Rightarrow S = \Omega$

  **Anonymity = 1**
Entropy

- Entropy $H(P)$ measures the “uncertainty” in a probability distribution
  
  $H(P) = -\sum_x P(x) \log_2 P(x)$

  Normalized: $H(P)/(\log_2 |\Omega|)$

- Example: $P$ is uniform on $S \subseteq \Omega$
  
  $H(P) = -|S| \sum (1/|S| \log_2 1/|S|) = \log_2 |S|$

- Example: $P(x) = 1$
  
  $H(P) = -\log_2 1 = 0$
Entropy

• Anonymity can be measured as expected entropy:
  – Anonymity \( \cdot E[\text{H(P)}] \)
  – Normalized: \( E[\text{H(P)}]/(\log_2 |\Omega|) \)

• Anonymity as bits:

  *How many more bits would the attacker need to identify the initiator with certainty?*
Routing to Attackers
Average Entropy = 0.60
Bad Distribution

Average Entropy = 0.30
Randomized Routing

- Chord routing: follow best (longest) finger that makes progress towards the destination
- Randomized routing: follow random finger that makes progress
  - Guaranteed to eventually arrive at destination
  - Performance bound grows to: $O(\log^2 N)$
  - Many more possible paths going through a node
Random Routing
Random Distribution

Average Entropy=0.70
Worst-Node Entropy=0.36
Random Distribution (Bad)

Average Entropy=0.67
Worst-Node Entropy=0.39
Performance Trade-Off

• Many possible optimizations to reduce path length
  – E.g. favor larger fingers when picking randomly
• Such optimizations lower path length
• But also lower the expected entropy
  – Not all paths are as likely
• Worst-case entropy suffers even more
Weighted Random Distribution

Average Entropy=0.55
Worst-Node Entropy=0.13
Realistic Distribution

Average Entropy=0.80
Worst-Node Entropy=0.25
High Indegree
Low Indegree
Summary

• Some users better off than others
  – High variance in expected entropy
• In-degree has a powerful effect on both attackers and honest participants
• In-degree influenced by relatively uncontrollable factors:
  – size of gap to predecessor
  – distance from destination
Intermediaries: Routing by proxy

- Route first to a random node $I$, then to the desired destination
- Really good for nodes in bad situations
  - e.g., low in-degree, attackers for neighbors
- How to get the routing request to the intermediary?
Intermediaries: Wrong Approach

(Mr. I, please route to D)
Intermediaries: Wrong Approach
Split Routing

• Use simple secret sharing to split up the message [D] ("please route to D")
  – generate R at random
  – split [D]!
    • $M_0 = [R]$
    • $M_1 = [R \otimes D]$
• $[D] = M_0 \otimes M_1$
• neither $M_0$ nor $M_1$ reveal D on their own
Split Routing

• \((D = M_0 \, \mathbb{C} \, M_1)\)

• Route \(M_0\) and \(M_1\) along *disjoint* paths to the intermediary \(I\)
  – attackers must intercept both messages
  – except at \(I\), paths have no chance of converging
  – probability of learning \(S!\) \(D\) is roughly *squared*
Bidirectional Routing

• How to route along disjoint paths?
• Standard routing is clockwise (CW)
  – fingers increase clockwise
    • ID + ½, ID + ¼, …
• Extend Chord to allow counterclockwise (CCW) routing as well
  – CCW fingers increase counterclockwise
    • ID – ½, ID - ¼, …
Bidirectional Routing

ID(S) + \(\frac{1}{4}\)

ID(S) + \(\frac{1}{2}\)

ID(S) - \(\frac{1}{4}\)

CW fingers

CCW fingers

S
Bidirectional Routing

- Can use randomized/etc. routing for each share
- Routing latency increased by ~2x
- Entropy harder to analyze
  - Must combine information from two attackers
  - Results will be in paper
- Can use more than 2 paths or more than 1 intermediary
  - Further trade-off performance and anonymity
Conclusion

• Analyzed anonymity properties of Chord and extensions
  – Identified properties that have the most effect on anonymity
  – Proposed and analyzed anonymity-enhancing modifications

• Our work can be generalized to other structured P2P systems
  – Indegree variations will be universally relevant
  – Intermediary/split routing can be used with other geometries