IP-to-Router Mapping Techniques and Results

Ken Keys, CAIDA
2008-02-13

The Alias Problem

- Traceroute reveals only one interface address on each router along a path.
- Given a set of IP paths, we can not tell which addresses belong to the same router.

Fingerprinting Solutions

- Send probe packets to different addresses, and identify similarities in responses that suggest they came from the same router.
- Accurate (low false positive rate)
- Not very complete (low true positive rate), because many routers do not respond to direct probes.

Analytical Solutions

- Draw inferences by analyzing the IP graph.
- Less accurate than fingerprinting
  - Depends on more assumptions about network engineering practice, heuristics, incomplete and sometimes conflicting data
- More complete than fingerprinting
  - Does not depend on direct probes

Common Source Address

- Send UDP or TCP packet to unused port at address A.
- If ICMP Port Unreachable response comes from address B, then A and B are aliases.
- Implementations: Mercator, iffinder

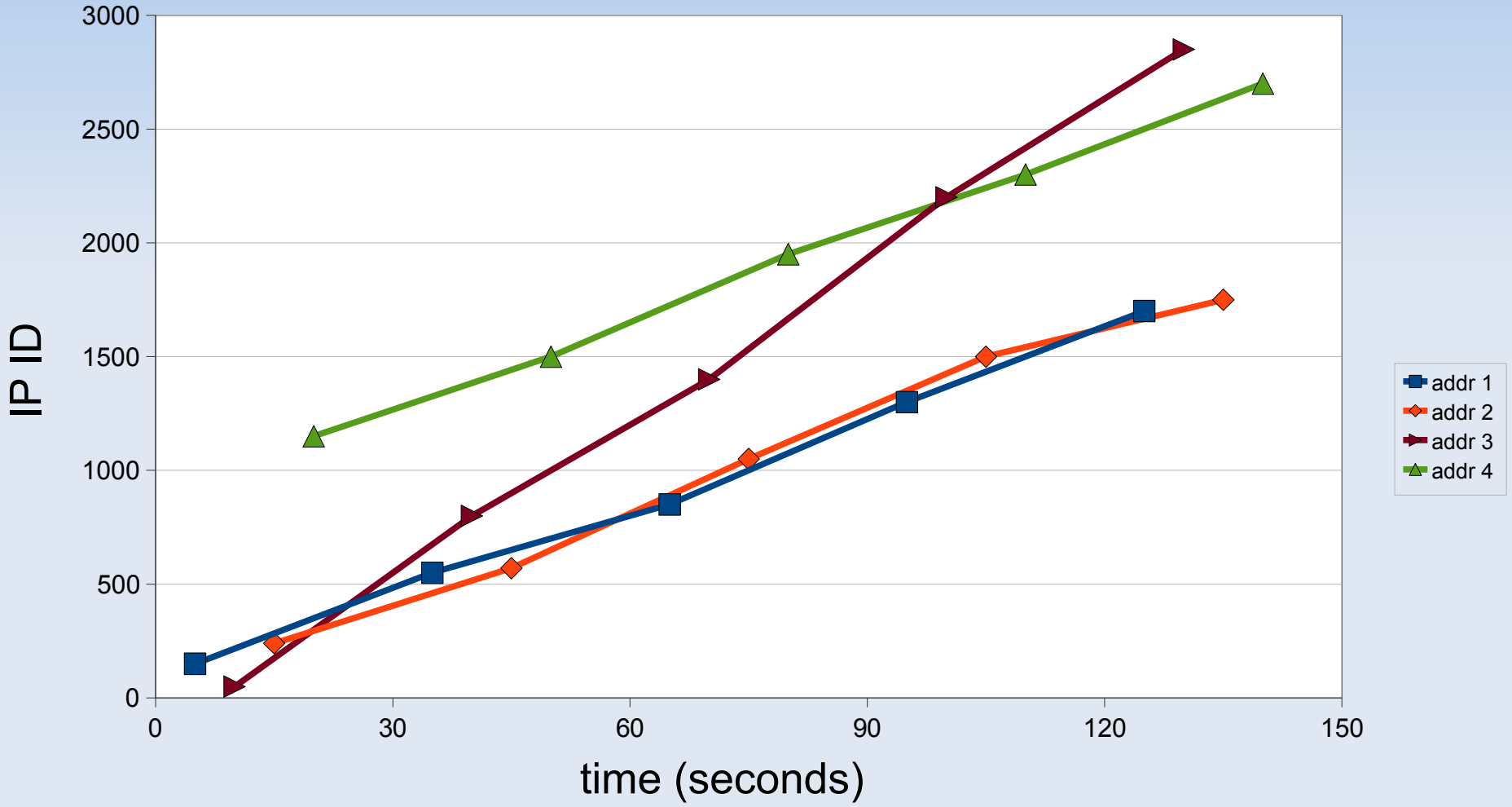
Common IP ID counter: Ally

- Many routers use a simple incrementing counter for the IP ID field.
- Ally sends packets to addresses A, B, A.
- If the responses have close ordered IP ID values, they may be from the same router.
- Problem: testing every possible (A, B) pair requires $O(n^2)$ probes.

Common IP ID counter: RadarGun

- Iterates over IP list multiple times, probing each address.
- Calculates “velocity”, or rate of change of IP ID counter over time, for each address.
- Any two addresses with similar velocity and predicted ID values are likely aliases.
- Improves upon Ally
  - Requires only *O(n)* probes
  - More tolerant of noise

RadarGun velocity example



DNS Analysis

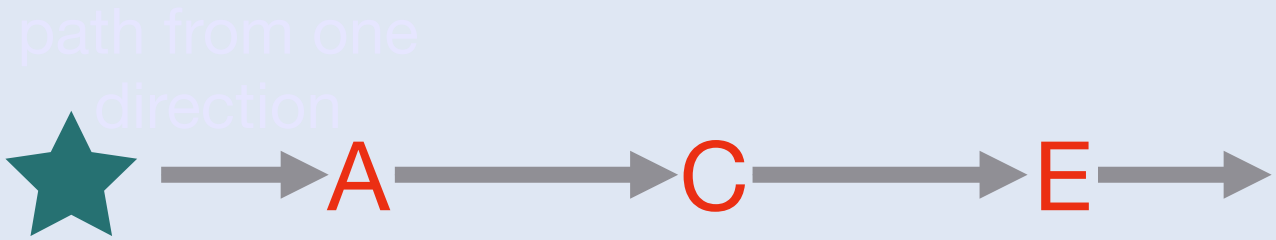
- Some organizations use DNS names for addresses that can be interpreted to identify aliases.
- Requires substantial human guidance.

Graph Analysis: APAR

- **A**nalytical and **P**robe based **A**lias **R**esolution
- Identify subnets among observed addresses.
  - Find common prefixes that do not cause contradictions (loops, broadcast addresses)
- Compare paths that cross the same subnets in opposite directions to infer aliases
- Optionally use TTL constraints to rule out false positives

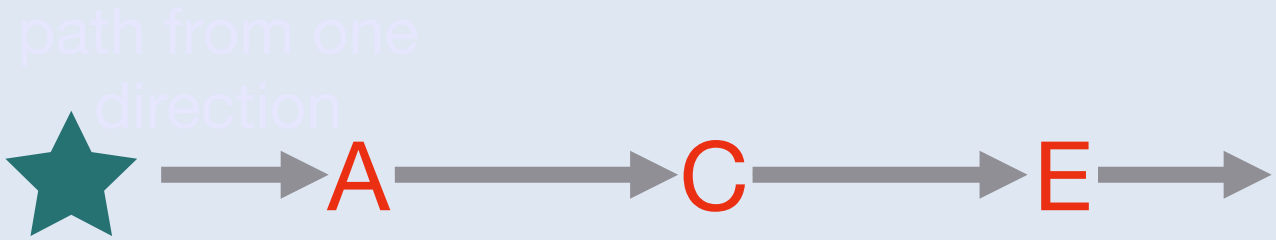
Graph Analysis: APAR

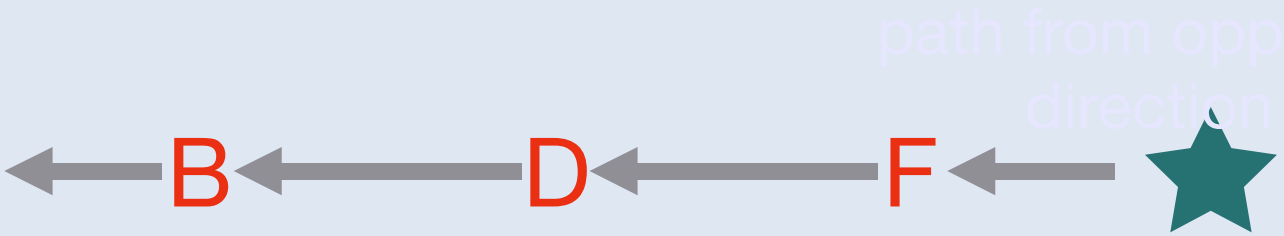
- Compare paths that cross the same subnets in opposite directions to infer aliases:



Graph Analysis: APAR

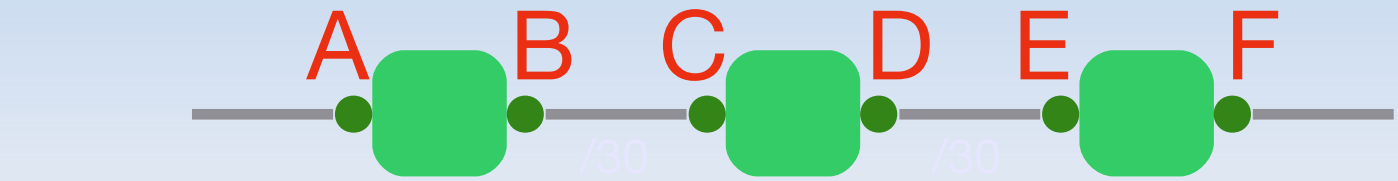
- Compare paths that cross the same subnets in opposite directions to infer aliases:

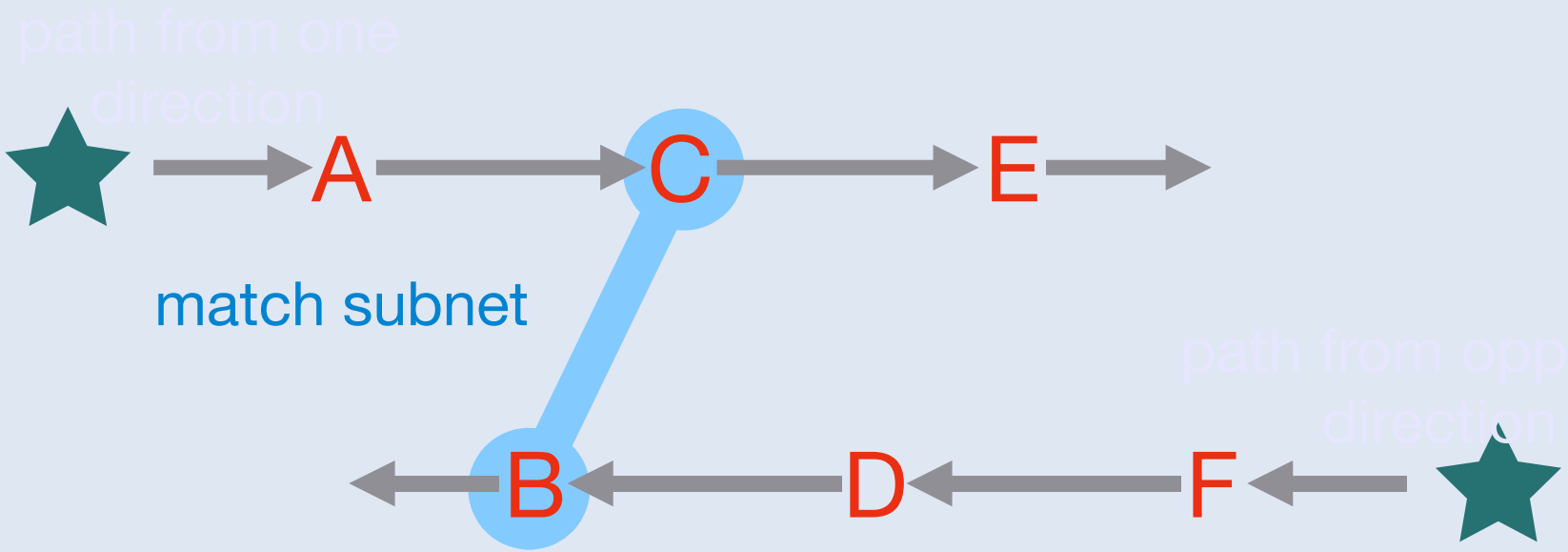




Graph Analysis: APAR

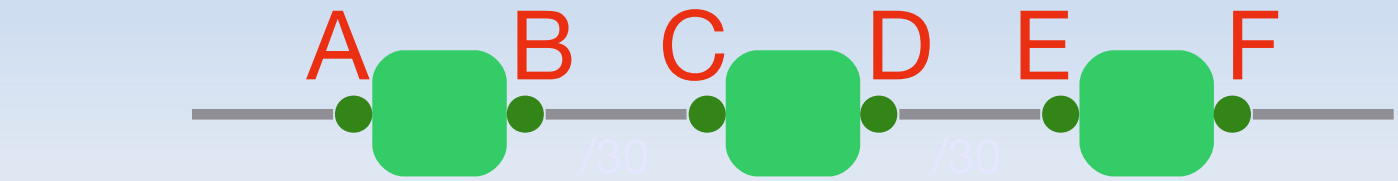
- Compare paths that cross the same subnets in opposite directions to infer aliases:

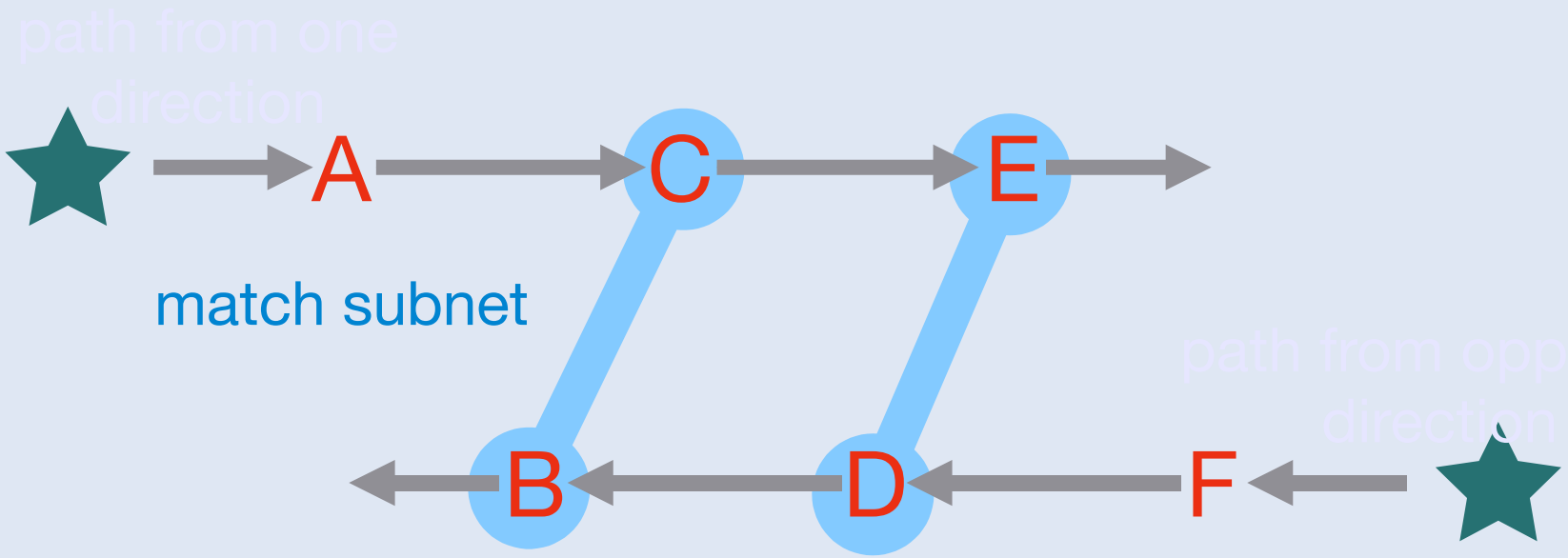




Graph Analysis: APAR

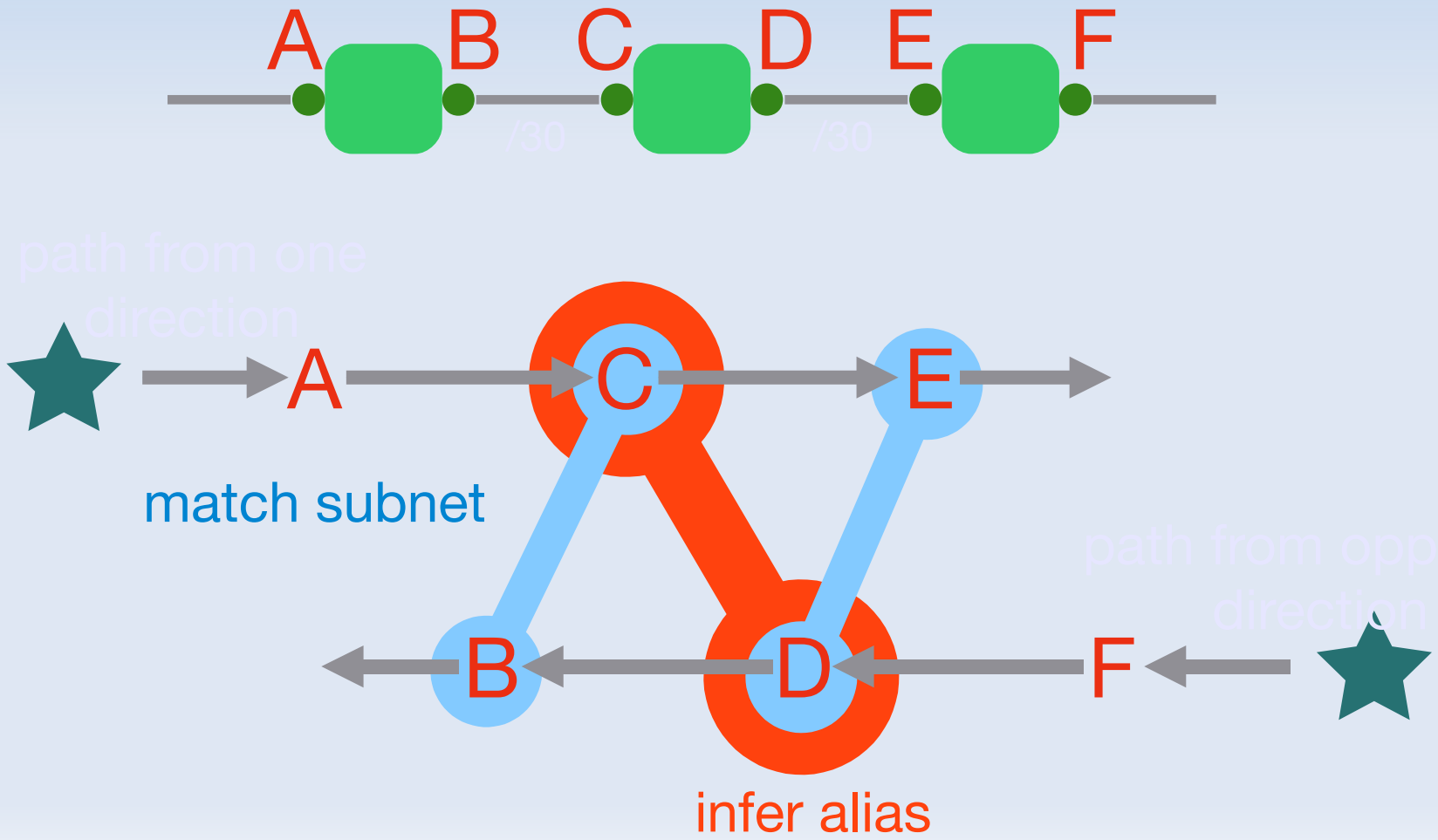
- Compare paths that cross the same subnets in opposite directions to infer aliases:





Graph Analysis: APAR

- Compare paths that cross the same subnets in opposite directions to infer aliases:



Graph Analysis: kapar

- Our implementation of the APAR algorithm
  - Optimized
  - Additional heuristics
    - TTLs from *multiple* vantage points
    - Stricter subnet inference rules
    - Additional probes to broadcast addresses of potential subnets

Graph analysis: DisCarte

- Combines traceroute data with Record Route data
- Uses Disjunctive Logic Programming to apply constraints and make inferences
- Extremely computationally expensive

Evaluation: data

- 373 M traceroutes from 26 Ark monitors
  - Found 2.4 M intermediate (router) addresses
  - Found 27 M total addresses
  - Ping each router address from all monitors, to collect TTLs
- Validated against known topology data from CANET, GÉANT, Internet2, NLR, and WIDE

Evaluation: results

	GEANT			Internet2			NLR		
	R	TP	FP	R	TP	FP	R	TP	FP
reality	18	540		9	713		7	231	
iffinder	0	0	0	0	0	0	6	100	0
kapar	14	75	6	15	193	26	9	61	7
kapar + TTL	11	80	6	12	163	6	8	67	6
iffinder + kapar	16	63	6	15	209	13	6	132	0
iffinder + kapar + TTL	11	84	6	14	167	4	7	127	0

R = routers with multiple interfaces
TP = true positive alias pairs
FP = false positive alias pairs

Evaluation: iffinder

- Ran on all 26 monitors to all router addresses
- Finds many aliases on networks where routers respond to direct probes, but finds no aliases on networks where routers do not respond
- Negligible false positive rate
- Using TTL constraints to check for false positives does more harm than good

Evaluation: APAR / kapar

- Works more evenly than iffinder across Internet
  - Finds 7 times as many alias pairs
- False positive rate is low, but significant
- Compared to APAR, kapar's stricter subnet rules and broadcast probes helped slightly
- TTL constraints reduce false positives (good), but also reduce true positives (bad); the net effect is a small benefit

Evaluation: iffinder + kapar

- Combines strengths of both methods
- In case of conflict, an iffinder alias is considered more reliable, because of iffinder's low false positive rate
- Even on parts of the Internet where iffinder does not find any aliases, results for iffinder+kapar are better than for kapar alone

Future work

- RadarGun
  - Still doesn't scale to CAIDA's IP graph
  - Using TTL-limited probes instead of direct probes should significantly improve response rate
  - Combine with iffinder and kapar
- TTLs
  - With multiple TTL probes, we hope to identify and discard inconsistent TTLs that hurt kapar's results

Thanks for listening

- Technical report available at http://www.caida.org/publications/papers/2008/alias_resolution_techreport/