

Building a Flexible and Efficient
Routing Infrastructure:
Need and Challenges

Karthik Lakshminarayanan

UC Berkeley

(with Ion Stoica and Scott Shenker)

IRIS Student Workshop, 2003

Overlays, Overlays, Overlays

- Why overlays?
 - Usually to get around the restrictive routing primitive of IP
 - Route quality, e.g. low loss rate routes
 - Quality of service
 - Multicast overlays (for different metrics of goodness)
 - And so on...
- Problem:
 - No/little synergy among the overlays

What can be shared?

- Higher level overlay functionality
 - Each application designs overlay routing from scratch
 - Lower deployment barrier: design effort & deployment expense
- Network weather information
 - Each application performs probes to find good overlay paths
 - Reduce overlay maintenance overhead

Application requirements

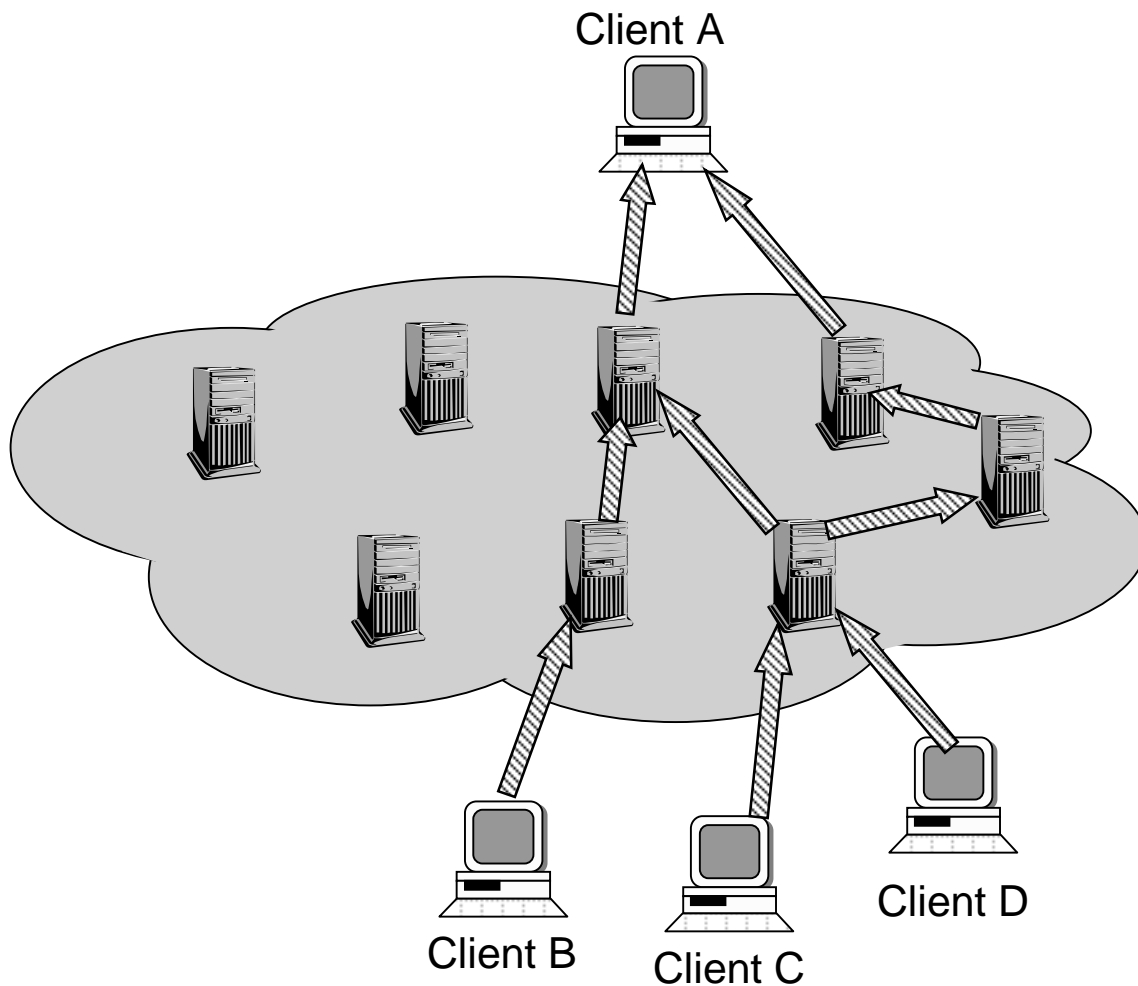
What are the requirements for supporting *most* of the overlays applications?

Give routing control to end hosts

- Adaptive routing: Based on *application sensitive metrics*
- Data manipulation: Ability to route to an intermediate node that manipulates/stores (e.g. transcode) data

Infrastructure exports *path abstraction*

➡ Setup routes

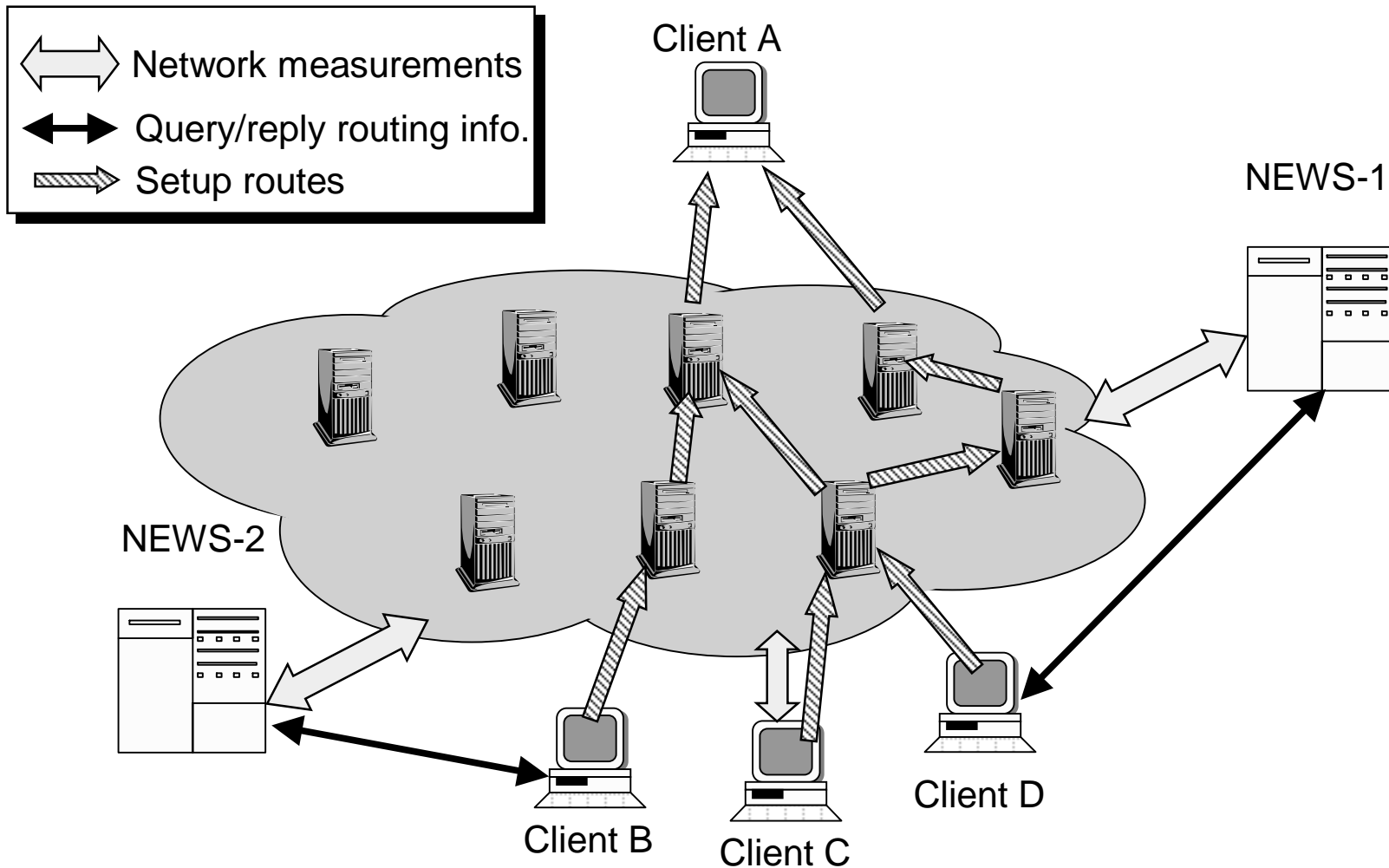


- Give end-hosts control over which path their packets can take
- Like a virtual circuit-based loose source routing primitive

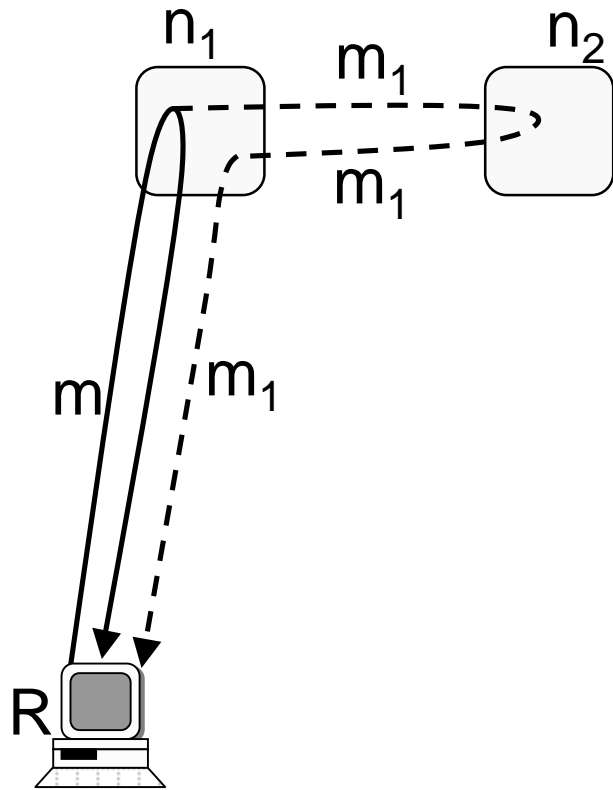
Challenges

- How do the applications know which paths to choose?
 - Third-party services perform measurements (of delay, loss, bandwidth) *using path abstraction alone*
 - Why not let infrastructure do measurement as well?
 - How do we do this measurement?
 - Information shared with overlay applications using an open interface
- How do we make loose source routing secure/robust?
 - Will not be covered in the talk

System architecture



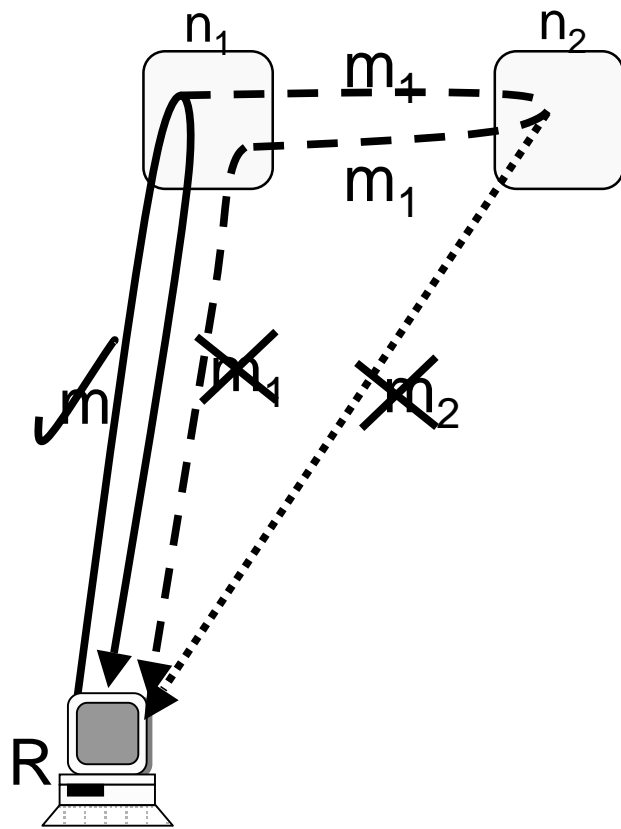
Measuring round-trip delay



To measure: $RTT(n_1 \rightarrow n_2)$

- Use path selection primitive to send packet m along $R \rightarrow n_1 \rightarrow R$
- Use path selection in conjunction with packet replication to send packet along $R \rightarrow n_1 \rightarrow n_2 \rightarrow n_1 \rightarrow R$
- Difference yields the RTT of the link $(n_1 \leftrightarrow n_2)$

Measuring one-way loss rate



To measure $l(n_1 \rightarrow n_2)$

- m_2 is used to differentiate loss on $(n_1 \rightarrow n_2)$ from that on $(n_2 \rightarrow n_1)$
- $(m \wedge \sim m_1 \wedge \sim m_2) \Rightarrow$ loss on virtual link $(n_1 \rightarrow n_2)$
 - False positives
 - False negatives
- Probability of false positives/negatives $\approx O(p^2)$

Measuring available bandwidth

- ...
- ...
- ...

Summary of design

- Infrastructure exports a path abstraction
- Delegate routing control to applications
 - Applications know their requirements best
- Delegate performance measurements to third-party applications
 - Allows this to evolve to meet changing requirement

Open questions & Future work

- Design: Why minimalist design?
- Measurements: what if path characteristics are correlated?
 - Shared bottleneck
 - Losses at the egress/ingress link
- Related problems:
 - By having incomplete information about network weather, how much do we lose (if at all)?
 - How much does accuracy of measurements affect the final outcome?
 - If the underlying routing is bad, what is the diversity of such an overlay needed to do a good job?
- Design/implement a P2P toolkit and develop applications based on it

That's All folks