A Framework for Highly-Available Cascaded Real-Time Internet Services

Bhaskaran Raman

Qualifying Examination Proposal

Feb 12, 2001

Examination Committee:

Prof. Anthony D. Joseph (Chair)

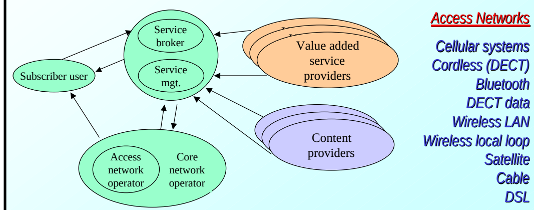
Prof. Randy H. Katz

Prof. Ion Stoica

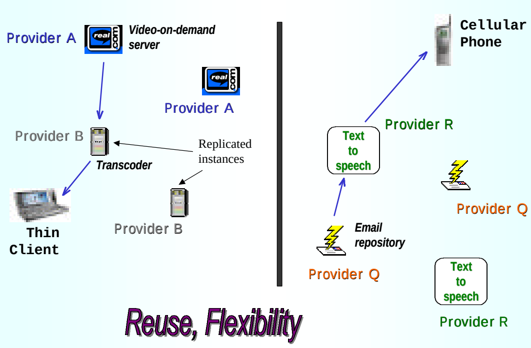
Prof. David Brillinger

Technological Trend

"Service and content providers play an increasing role in the value chain. The dominant part of the revenues moves from the network operator to the content provider. It is expected that value-added data services and content provisioning will create the main growth."



Service Composition



Service Composition

- Quick development and deployment
- Notion of service-level path
- Composition across
  - Service providers
  - Wide-area

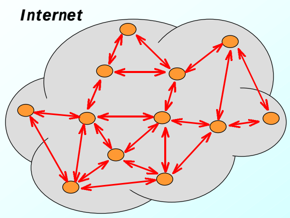
Requirements and Challenges

- Framework for composing services
  - How are services deployed/replicated?
  - Who composes services? How are service-level paths created?
- Choice of "optimal" service-level path
- Robustness
  - Detect and recover from failures

Requirements and Challenges: High Availability

- Services need to be available always
- Important for long-lived sessions
  - Several minutes or hours
  - Should be uninterrupted
- Availability in PSTN: 99.999%
  - Backup during session
- Internet:
  - 10% of paths have less than 95% availability
  - BGP convergence could take several minutes

Overall Approach



- Overlay of clusters
  - Service platform
- Peering
  - Cascade services
  - Active monitoring
- Connection-oriented network
  - Highly available service-level paths

Problem Scope

- Services have no "hard" state
  - Sessions can be transferred from one service instance to another
  - This is assumed while handling failures
- Assumption valid for a large set of applications

Outline

- Related work
- Feasibility of failure detection over the wide-area
- Design of the framework
- Evaluation
- Research methodology and timeline
- Summary

Related work

- Composition of services
- Robustness and performance for Internet services
- Overlay networks on the Internet
- Routing around failures

Related work

- Service composition
  - TACC
    - Within a single service-provider cluster
    - Model does not extend for composition across clusters
  - Simja, COTS
    - Semantic issues addressed
    - Also does not address composition across service providers
    - Do not address availability or performance issues
    - This work is complementary to ours

Related work

- Fault tolerant Internet services
  - TACC, AS1
    - Web-proxy, Video-proxy
    - Fault-tolerance within cluster, not across
    - Not in the context of wide-area composed paths
  - LARD
    - Web-server fault-tolerance
    - Selection of instance within cluster
  - Mirror selection (e.g. SPAND)
    - Not for composed services
    - No recovery *during* a long-lived session

Related work

- Overlay networks
  - Tapestry, CAN
    - Locate replicated objects in the wide-area
  - Intentional Naming System
    - Overlay to route based on service descriptions
  - Resilient Overlay Networks
    - Routing around network failures
- Key differences
  - Provision for service composition
  - Connection-oriented overlay network

Related work

- Routing around failures in networks
  - BGP
  - MPLS, ATM
- Idea of peering between service clusters similar to BGP peering
- We can borrow different flavors of recovery from connection-oriented networks
  - Backup paths
  - Routing around failed link
- But, we have to adapt these to composed services

Related work: summary

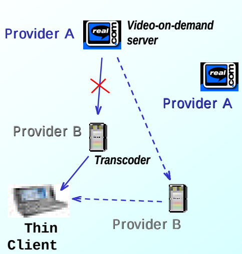
Composed services ➔ composition across the network not addressed

High availability & performance issues addressed ➔ not in the context of composed services

Outline

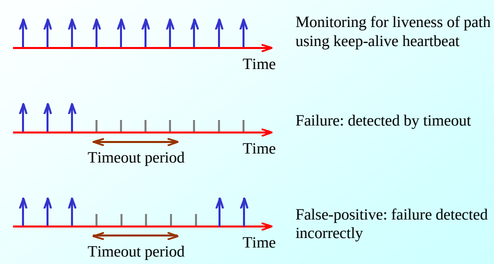
- Related work
- Feasibility of failure detection over the wide-area
- Design of the framework
- Evaluation
- Research methodology and timeline
- Summary

Failure detection in the wide-area: Analysis



- Need to detect failures during a session
- Given Internet cross-traffic, congestion, is this possible?
- Need an idea of the false-positive rate

Is Wide-Area Monitoring Feasible?



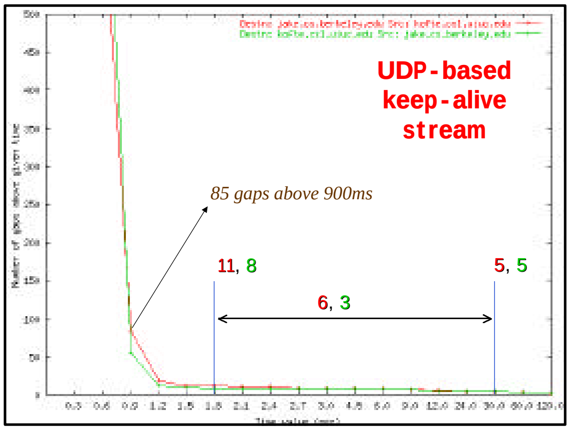
There's a trade-off between time-to-detection and rate of false-positives

Is Wide-Area Monitoring Feasible?

- False-positives due to:
  - Sudden increase in RTT
  - Simultaneous losses
- Previous studies:
  - Internet RTT study, Acharya & Saltz, UMD 1996
    - RTT spikes are isolated; undone in a couple of seconds
  - TCP RTO study, Allman & Paxson, SIGCOMM 1999
    - Significant RTT increase is quite transient
    - 86% of bad TCP timeouts are due to one or two elevated RTTs
  - Our experiments using ping servers
    - Loss-runs > 4 are very rare; once in an hour to once in a day
- We perform UDP-based heartbeat experiments

UDP-based keep-alive stream

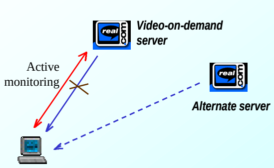
- Geographically distributed hosts:
  - Berkeley, Stanford, UIUC, TU-Berlin, UNSW
- UDP heart-beat every 300ms
  - Represents low rate
  - Internet jitter is of the order of 100ms anyway
- Measure gaps between receipt of successive heart-beats



UDP Experiments: Summary

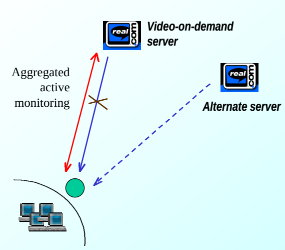
- False-positive rate is quite low
  - As low as 50%
- Good amount of variability, but many host pairs are "well-behaved"
- Failures do happen in the end-to-end path
  - About once in a day
  - Much higher failure rate than telecommunication network

Design Alternative: End-to-end monitoring



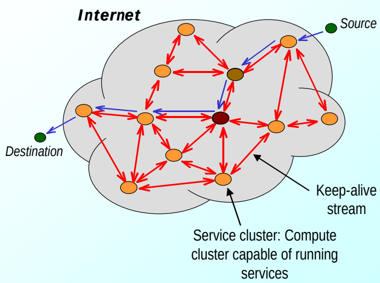
- No infrastructure required
  - Hop-by-hop composition
- Problems:
  - Overhead
  - Sub-optimal service-level path
  - Alternative path may not be active
  - What if both ends are fixed?

Design Alternative: Client-Side Aggregation



- Reduces overhead
- Other problems persist:
  - Hop-by-hop composition
  - Alternate server could be unavailable
  - Does not work if both ends are fixed

Architecture



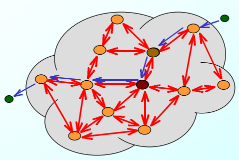
Middleware service platform

Service-level paths through this overlay network

Architecture

- Overlay nodes are clusters
  - Hierarchical monitoring
    - Within cluster – for process/machine failures
    - Across clusters – for network path failures
  - Aggregated monitoring
    - Amortized overhead
- Overlay network
  - Context for exchanging information to form service-level paths
  - Intuitively, expected to be much smaller than the Internet
  - With nodes near the backbone, as well as near edges

Routing on the overlay network



- Find *entry* point and *exit* point
- Find route in the overlay network, through services
- Mechanism for recovery from failures

Routing: Finding entry and exit

- Independent of other mechanisms
- Entry or exit point can be rather static
  - Nodes are clusters ➔ do not fail often
  - By placement, can make choice of overlay node obvious
- Can learn entry or exit point through
  - Pre-configuration,
  - Expanding scope search,
  - Or, any other level of indirection

Routing on the overlay network

- Connection-oriented network
  - Explicit session setup stage
  - There's "switching" state at the intermediate nodes
- Need a connection-less protocol for connection setup
- Need to keep track of three things:
  - Network path liveness
  - Metric information (latency/bandwidth) for optimality decisions
  - Where services are located

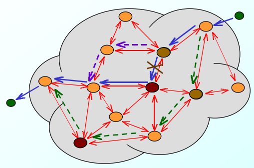
Routing on the overlay network

- Three levels of information exchange
  - Network path liveness
    - Low overhead, but very frequent
  - Metric information: latency/bandwidth
    - Higher overhead, not so frequent
    - Bandwidth changes only once in several minutes
    - Latency changes appreciably only once in about an hour
  - Information about location of services in clusters
    - Bulky
    - But does not change very often (once in a few weeks, or months)
- Link-state algorithm to exchange information
  - Least overhead ➔ max. frequency
- Service-level path created at entry node

Routing on the overlay network

- Two ideas:
  - Path caching
    - Remember what previous clients used
    - Another use of clusters
  - Dynamic path optimization
    - Since session-transfer is a first-order feature
    - First path created need not be optimal

Recovery in the overlay network



- End-to-end vs. local-link
  - Pre-established vs. on-demand
  - Can use a mix of strategies
- Pre-established end-to-end:
  - Quicker setup of alternate path
  - But, failure information has to propagate
  - And, performance of alternate path could have changed
- On-demand local-link:
  - No need for information to propagate
  - But, additional overhead

The Overlay Topology

- Need to address:
  - How many overlay nodes are deployed?
  - Where are they deployed?
  - How do they decide to peer?

The Overlay Topology

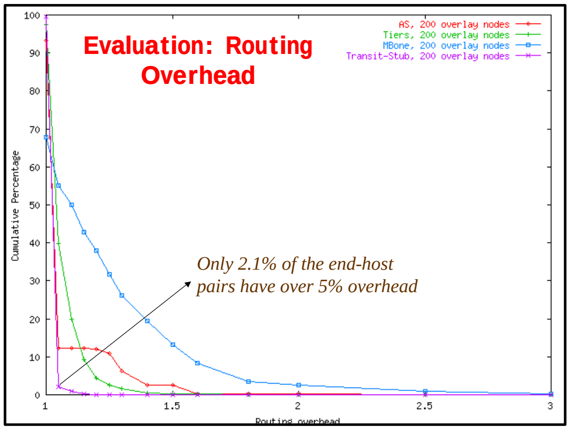
- How many nodes?
  - Large number of nodes ➔ lesser latency overhead
  - But scaling concerns
- Where to place nodes?
  - Need to have overlay nodes close to edges
    - Since portion of network between edge and closest overlay node is not monitored
  - Need to have overlay nodes close to backbone
    - Take advantage of good connectivity
- Who to peer with?
  - Nature of connectivity
  - Least sharing of physical links among overlay links

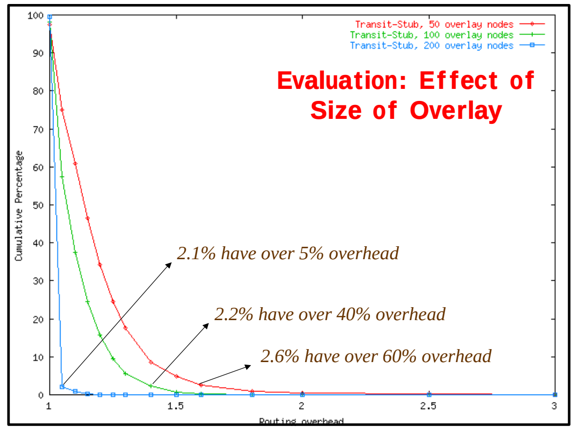
Outline

- Related work
- Feasibility of failure detection over the wide-area
- Design of the framework
- Evaluation
- Research methodology and timeline
- Summary

Evaluation

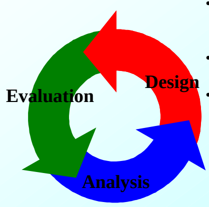
- Important concern: overhead of routing over the overlay network
  - Addition to end-to-end latency
- Network modeling
  - AS-Jan2000, MBone, TIERS, Transit-Stub
  - Between 4000-6500 nodes
- Overlay nodes
  - 200: those with max. degree (backbone placement)
  - Peering between “nearby” overlay nodes
  - Physical links are not shared





Research Methodology

- Simulation
  - Routing overhead
  - Effect of size of overlay
- Implementation
  - Codec service for IP-telephony
  - MP3 music for GSM cellular-phones



- Connection-oriented overlay network of clusters
- Session-transfer on failure
- Aggregation – amortization of overhead

- Wide-area monitoring trade-offs
  - How quickly can failures be detected?
  - Rate of false-positives

Research Methodology: Metrics

- Overhead
  - End-to-end latency; bandwidth for information exchange
- Latency to recovery
  - Measure of effectiveness
- Use of composability
  - For building application functionality
- Scalability
  - To a large number of client sessions
- Stability
  - Of optimality and session-transfer decisions

Research Methodology: Approach

- Simulations, Trace-collection, Real implementation
- Simulation
  - For initial estimation of overhead
- Simulation + Traces
  - Bandwidth usage estimation, Stability study
- Real implementation
  - Scalability studies
  - Real services for use of composability
- Testbed
  - Collaborating with UNSW, TUBerlin

Research Plan: Phase I (0-6 months)

- Detailed analysis of
  - Latency and bandwidth overhead
  - Latency to recovery
- Use traces of latency/bandwidth over wide-area
- Develop real implementation in parallel
  - This is already in progress
  - Will give feedback for the analysis above

Research Plan: Phase II (6-12 months)

- Use implementation from Phase I
  - Deploy real services on the wide-area testbed
  - Analyze end-to-end effects of session-recovery
  - Examine scalability
- Use traces from Phase I to analyze stability of optimality decisions
  - Collect more traces of latency/bandwidth

Research Plan: Phase III (12-18 months)

- Use feedback from deployment of real services to refine architecture
- Analyze placement strategies
  - Use wide-area measurements and traces from phases I and II
- Write, graduate…

Appropriate conferences and workshops: NOSSDAV, ACM Multimedia, SOSP, INFOCOM, SIGCOMM

Summary of Contributions

- Framework for service composition across service providers
- Notion of connection-oriented network at the service-level
  - For optimizing paths
  - For detecting and recovering from failures