ECHO: Recreating Network Traffic Maps for Datacenters with Tens of Thousands of Servers

Christina Delimitrou\textsuperscript{1}, Sriram Sankar\textsuperscript{2}, Aman Kansal\textsuperscript{3}, Christos Kozyrakis\textsuperscript{1}

\textsuperscript{1}Stanford University
\textsuperscript{2}Microsoft
\textsuperscript{3}Microsoft Research

IISWC – November 5\textsuperscript{th} 2012
Motivation

Network Performance and Efficiency → critical for DC operation

- **Scalable Topologies**
  - Dragonfly, Fat tree, Clos, etc.
  - Hotspot detection & elimination

- **Flow Control**
  - Load balancing
  - Speculative flow control
  - Hedera, etc.

- **Network Switches Design**
  - Low latency RPCs
  - RAMCloud, etc.

- **Software-defined DC networks**
  - OpenFlow
  - Nicira, etc.
Challenge

Where to find representative traffic patterns??
Executive Summary

- **Network Workload Model:** A scheme that accurately and concisely captures the traffic of a DC workload
  - User patterns only emerge in large-scale → scalability
  - Different level of detail per application → modularity/configurability

- Prior work on network modeling → mostly single-node, temporal behavior
  - No spatial patterns, scalability and modularity

- **ECHO** addresses limitations of previous schemes:
  - **System-wide network modeling:** Not confined to a single-node
  - **Locality-aware:** Accounts for spatial network traffic patterns
  - **Hierarchical:** Adjusts the level of granularity to the needs of each app/study
  - **Scalable:** Scales to DCs with ~30,000 servers
  - **Lightweight:** Low and upper-bound modeling overheads
  - **Validated:** ECHO is validated against real traces from applications in production DCs
Outline

- Simple Temporal Model
- DC Network Traffic Characterization
- ECHO Design
- Model Validation
Distribution Fitting Model

- Most well-known modeling approach for network
- Single-node as opposed to system-wide!
- Capture temporal patterns in per-server network traffic
- Identify known distributions (e.g., Gaussian, Poisson, Zipf, etc.) in network activity traces
- Represent server network activity as a superposition of identified distributions

\[ BW = \sum_{i=0}^{N} Distributions = \sum_{i=0}^{N_1} \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{1}{2} \left( \frac{x - \mu_i}{\sigma_i} \right)^2} |t_{i\text{stop}} - t_{i\text{start}}| + \]

\[ \sum_{i=0}^{N_2} \frac{\lambda^k e^{-\lambda}}{k!} |t_{i\text{stop}} - t_{i\text{start}}| + \sum_{i=0}^{N_3} f_{others} |t_{i\text{stop}} - t_{i\text{start}}| + \ldots \]
Distribution Fitting Model

- Capture temporal patterns in per-server network traffic
- Identify known distributions (e.g., Gaussian, Poisson, Zipf, etc.) in network activity traces
- Represent server network activity as a superposition of identified distributions

Model = Gaussian + Exponential + Gaussian + Gaussian + Constant

Validation: Deviation between original and synthetic is 4.9% on average
Distribution Fitting Model

Positive:
✓ Simple, accurate and concise
✓ Captures temporal patterns in network activity
✓ Facilitates traffic characterization (traffic is expressed as well-studied distributions)

Negative:
✗ Does not track spatial patterns
✗ Bursts in network activity not easily emulated by known distributions → would complicate the model
✗ Non-modular design
Outline

- Simple Temporal Model
- DC Network Traffic Characterization
- ECHO Design
- Model Validation
Methodology

- **Workloads:**
  - Entire *Websearch* application
  - *Combine* $\rightarrow$ Websearch query results aggregator
  - *Render* $\rightarrow$ Websearch query results display

- **Experimental systems are production DCs with:**
  - 30,000 servers running *Websearch*
  - 360 servers running *Combine*
  - 1350 servers running *Render*

- We collect per-server bandwidth traces of data sent and received over a period of 5 months (at 5msec granularity)
Understanding Network-wide Behavior

- **Temporal variations of network traffic**
  - Fluctuation over time
  - Differences between workloads

- **Average spatial patterns in network activity**
  - Locality in network traffic
  - Impact of application functionality to locality

- **Temporal variations in spatial patterns**
  - Changes over different time scales
  - Changes for different types of workloads
Temporal Variations in Network Traffic

- Most servers are greatly underutilized → significant overprovisioning for latency-critical apps
- Some servers have higher utilization → mostly well load-balanced
- Similarity in network activity patterns over time
- Model should: capture fluctuation, remove information redundancy
Temporal Variations in Network Traffic

- Clearer diurnal patterns → 31 dark and 31 light vertical bands
Temporal Variations in Network Traffic

- Clearer diurnal patterns $\Rightarrow$ 31 dark and 31 light vertical bands
- Higher utilization $\Rightarrow$ not as much overprovisioning for servers that aggregate query results
Temporal Variations in Network Traffic

- Clearer diurnal patterns \(\rightarrow\) 31 dark and 31 light vertical bands
- Higher utilization \(\rightarrow\) not as much overprovisioning for servers that aggregate query results
- Not equally load-balanced \(\rightarrow\) impact of queries serviced by each server
Spatial Patterns in Network Activity

- High spatial locality $\Rightarrow$ Most accesses are confined within the same rack
- The model should preserve the spatial locality (within racks & hotspots)
Spatial Patterns in Network Activity

- High spatial locality $\Rightarrow$ Most accesses are confined within the same rack
- The model should preserve the spatial locality (within racks & hotspots)
- A few servers communicate with most of the machines $\Rightarrow$ cluster scheduler, aggregators, monitoring servers
Spatial Patterns in Network Activity

- In contrast, Combine has less spatial locality → most servers talk to many machines
- Consistent with its functionality → query aggregation
Fluctuations in Spatial Patterns

- At first glance spatial locality is very similar across months
Fluctuations in Spatial Patterns

- At first glance, spatial locality is very similar across months.
- However, at finer granularity, there are differences.
### Fluctuations in Spatial Patterns

At first glance, spatial locality is very similar across months. However, at finer granularity, there are differences:

- Software updates
- Changes in traffic due to user load
- Background processes (e.g., garbage collection, logging, etc.)
Fluctuations in Spatial Patterns

- At first glance spatial locality is very similar across months
- However, at finer granularity there are differences
  - Software updates
  - Changes in traffic due to user load
  - Background processes (e.g., garbage collection, logging, etc.)
- Fine-grain patterns important for studies focused on specific hours of the day
Outline

- Simple Temporal Model
- DC Network Traffic Characterization
- ECHO Design
- Model Validation
Model Requirements

Don’t just model a node. Model the whole DC!

Requirements:

1. **Average** activity over time and space
2. **Per-server activity** fluctuation over time
3. **Spatial** patterns in network traffic
4. **Individual** server-to-server communication
Hierarchical Markov Chain: groups of racks $\rightarrow$ racks $\rightarrow$ individual servers

Configurable granularity based on app/study requirements

Captures spatial patterns in network traffic: fine-grain transitions are explored within each coarse state $\rightarrow$ most locality confined within a rack
Model Design – Temporal Aspects

- Captures temporal patterns in network traffic → multiple models used over time
- Number of models is a function of the workload’s activity fluctuations
- Switching between models allows compression in replay → fast experimentation
Hierarchical vs. Flat Model

- **Hierarchical**: explore fine grain transitions within coarse states
- **Flat**: explore all fine grain states → exponential increase in transition count
- Even for problems with a few hundred servers the model becomes intractable
- No loss in accuracy with the hierarchical model since locality is mostly confined within racks
Model Construction

- Collect system-wide network activity traces
- Cluster network requests based on
  - Sender/receiver server IDs
  - Type (rd/wr) and size of request (MB)
  - Inter-arrival time between requests (ms)
- Compute transition probabilities between states (e.g., S1 → S2: 90% 8KB read requests, 10msec inter-arrival time)
Focus on specific interesting activity patterns → Validating the model in server subsets (a few hundred servers)

Network activity is not necessarily self-contained in those server subsets

Cloud Node: Emulate all network activity to and from servers external to the studied server subset

Maintains accuracy of per-server load while enabling more fine-grain validation
Outline

- Simple Temporal Model
- DC Network Traffic Characterization
- ECHO Design
- Model Validation
Validation

1. Temporal variations of network activity
2. Spatial patterns of network activity
3. Individual server interactions (one-to-one communication patterns)
Less than 8% deviation between original and synthetic workload, on average across server subsets
Validation – Spatial Patterns

Less than 10% deviation between original and synthetic workload, on average across server subsets
Validation – Indiv. Server Interactions

- 12% deviation between original and synthetic for a weekday
- 9% deviation between original and synthetic for a day of the weekend
Validation – Benefits of Hierarchy

1 Level: 28% deviation
2 Levels: 9.1% deviation
3 Levels: 4.4% deviation
Motivation: Revisited

- **Scalable Topologies** ✓
  - Dragonfly, Fat tree, Clos, hotspot detection & elimination

- **Flow Control** ✓
  - Load balancing
  - Speculative flow control, Hedera, etc.

- **Network Switches Design** ✓
  - High port count designs, low latency RPCs, RAMCloud, etc.

- **Software-defined DC networks** ✓
  - OpenFlow, Nicira, etc.

- **Security attacks** ✓
  - Real-time deviation from modeled behavior

- **Retraining for major sw updates, major system configuration changes**
  - Low overhead process
Conclusions

- ECHO leverages validated analytical models to capture the temporal and spatial access patterns in DC network activity.
- It preserves the intensity and characteristics of DC network traffic.
- It adjusts the granularity of representation to the app/study demands.
- It is scalable and lightweight.
- Decouples network system studies from access to large-scale applications.

Future work

- Use ECHO for network system studies without the requirement for full application deployment.
- Expand similar concepts to other subsystems.
Questions??

Thank you