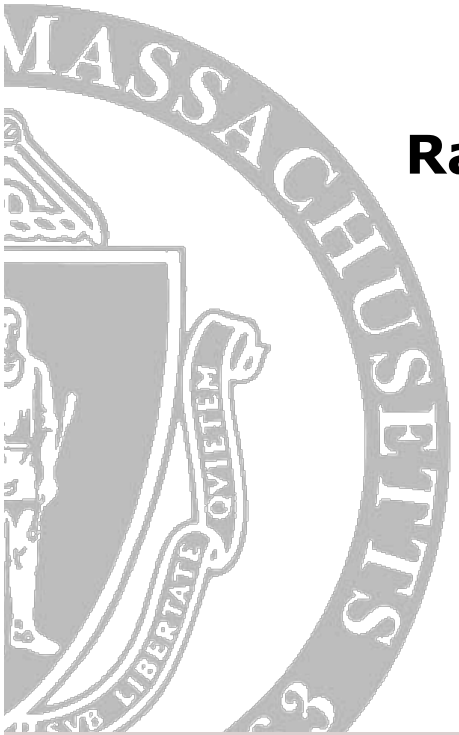
UMassAmherst

Yank: Enabling Green Data Centers to Pull the Plug



Rahul Singh, David Irwin, Prashant Shenoy
University of Massachusetts Amherst

K.K. Ramakrishnan
AT&T Research

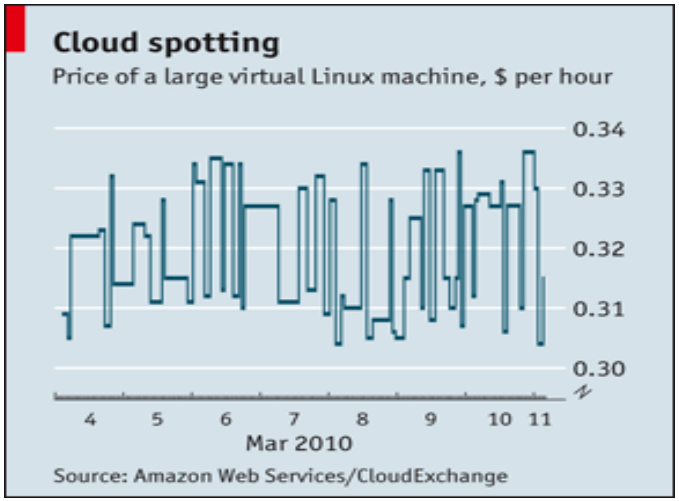
Data Center Reliability

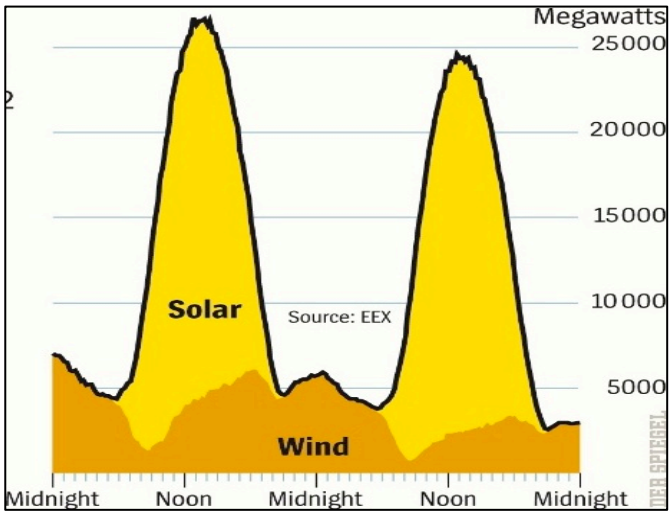
- Infrastructure designed to be highly available
  - Applications expect stable servers
    - Highly redundant power infrastructure
  - Availability is expensive

- Alternative approach
  - Relax strict stability assumption
  - Design low-cost HA techniques to compensate

Transient Servers

- Transiency Scenarios
  - Spot instances in Amazon Cloud
  - Server downtime due to power outage
  - Use of intermittent renewables
- **New abstraction:** Transient Server
  - Unpredictable availability
  - Receive advance warning of termination





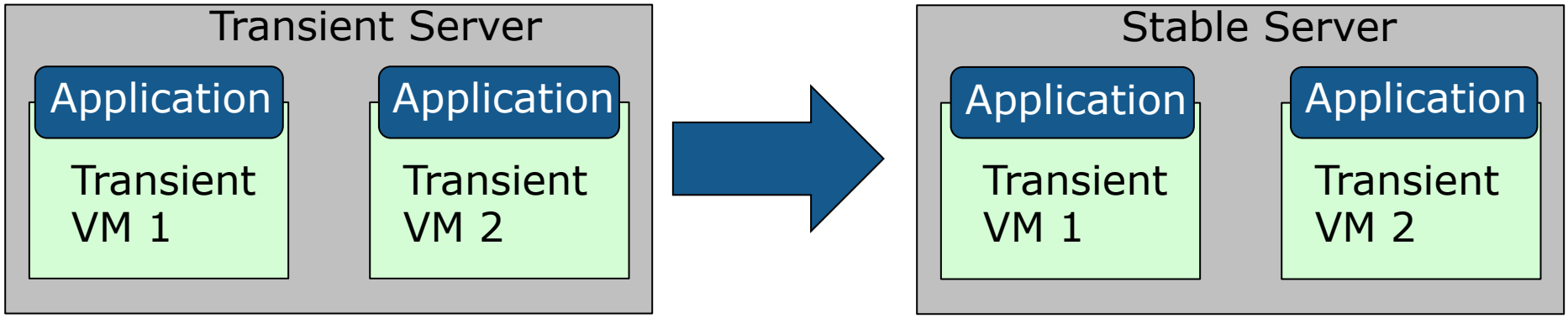
Yank Problem Statement

How to maintain application availability while allowing data centers to transparently use transient servers?

- Introduce Transient Server Abstraction
- Design Yank: System Support for Transient Servers
- Apply Yank to Green Data Centers

Supporting Transient Servers

- Two Ways of Using Transient Servers
  - Modify Application
    - *Easy* – Batch Applications
    - *Hard* – Interactive (Web) Applications
  - System Support
- Yank: System Support for Transiency
  - Given warning, transfer transient VMs to stable server
  - Must complete transfer within warning time



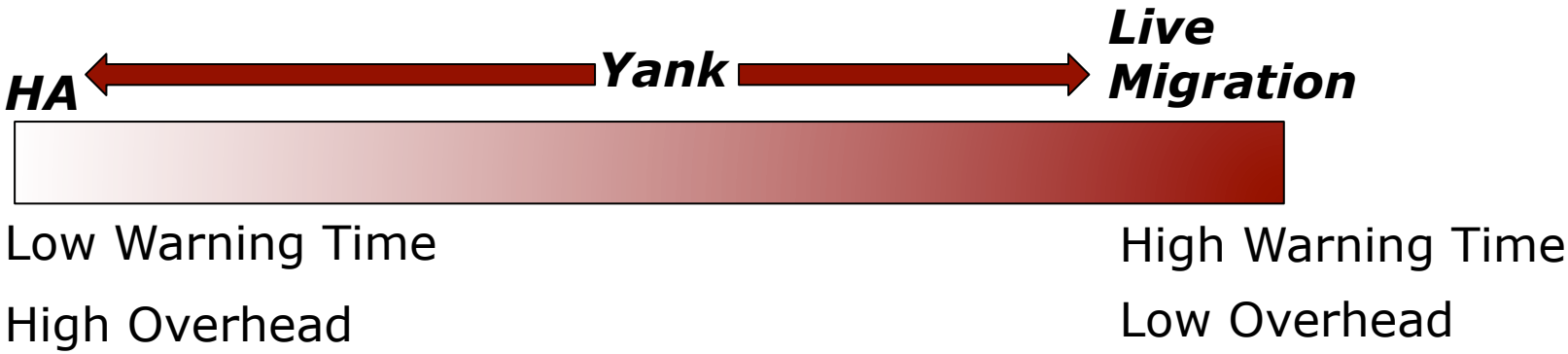
Strawman Approaches

- Live Migration
  - Xen Live Migration [NSDI 2005]
  - *Pros:* Low Overhead
  - *Cons:* Large Warning Time (~70-100secs)

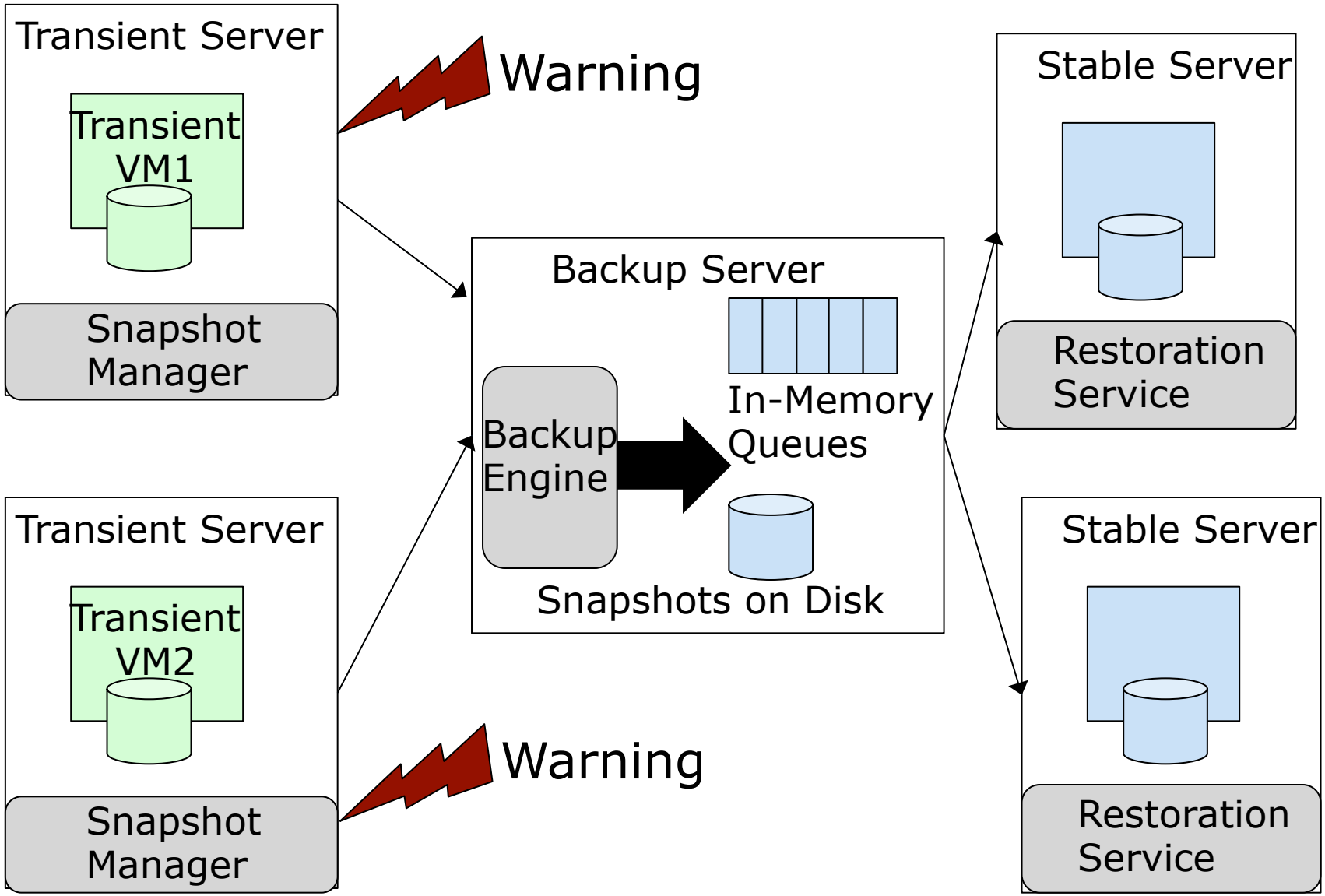
- High Availability (HA) Solutions
  - Remus [NSDI 2008]
  - *Pros:* Low Warning Time
  - *Cons:* High Overhead, High Hardware Cost

Yank's Approach

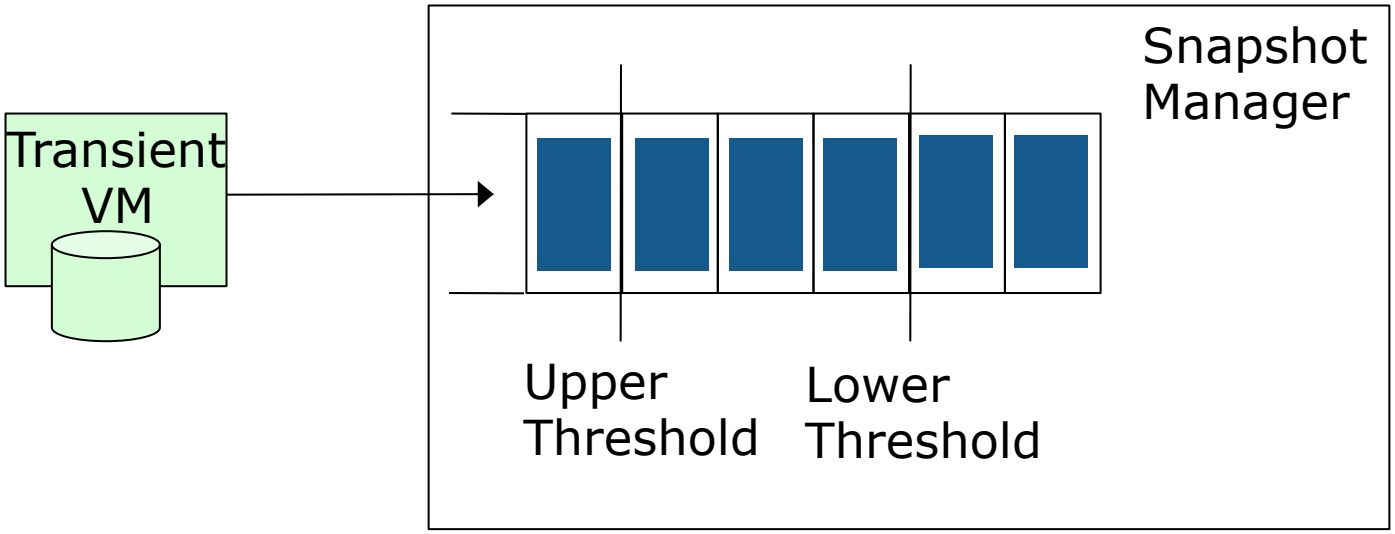
- Yank Covers Entire Spectrum of Warning Time
  - Low Warning Time -> Equivalent to HA
  - High Warning Time -> Equivalent to Live Migration
- Adapts to Warning Time
  - Overhead depends on the warning time



Yank High-Level Design

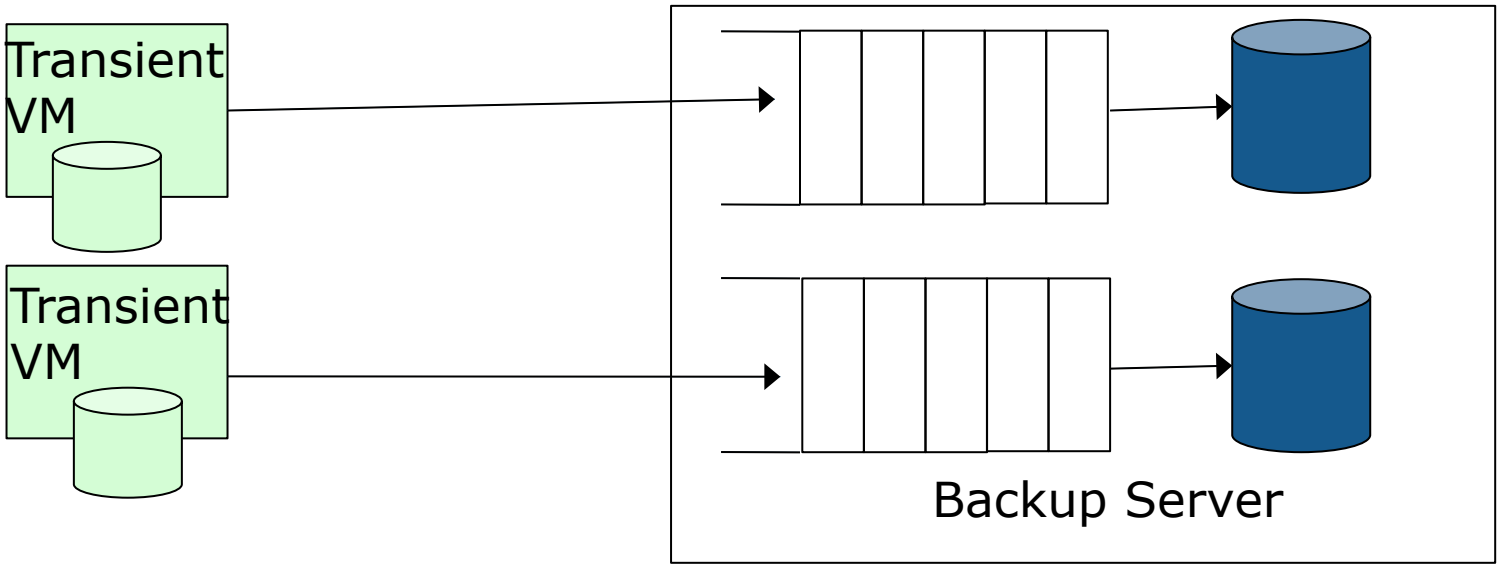


Snapshot Manager



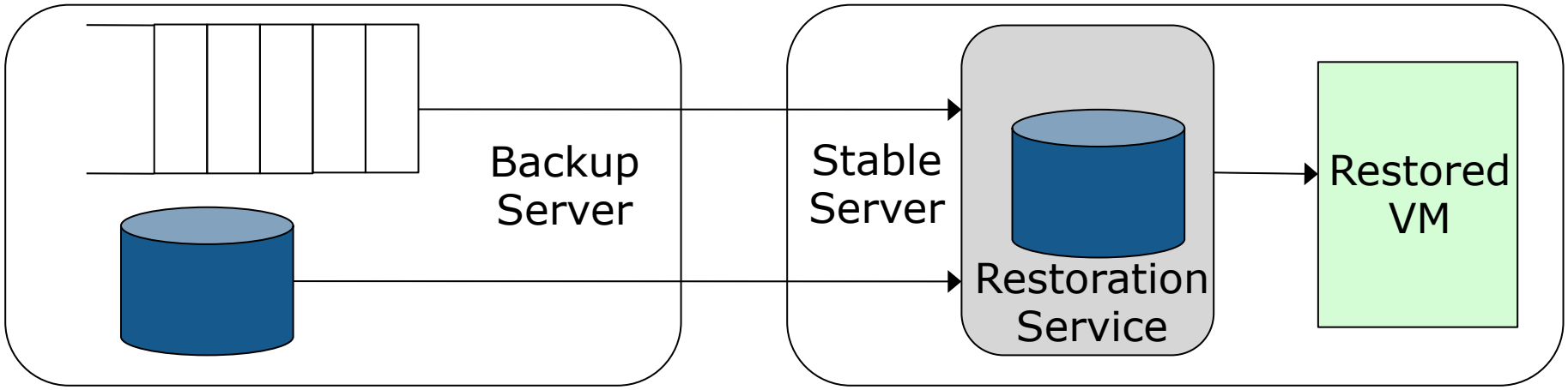
- Limit on dirty state sent within warning time
- **One option:** one threshold
  - Send when *[size of dirty state < limit]*
- **Alternative:** two thresholds
  - Upper threshold -> Synchronous send with buffering
  - Lower threshold -> Asynchronous send with no buffering

Backup Engine



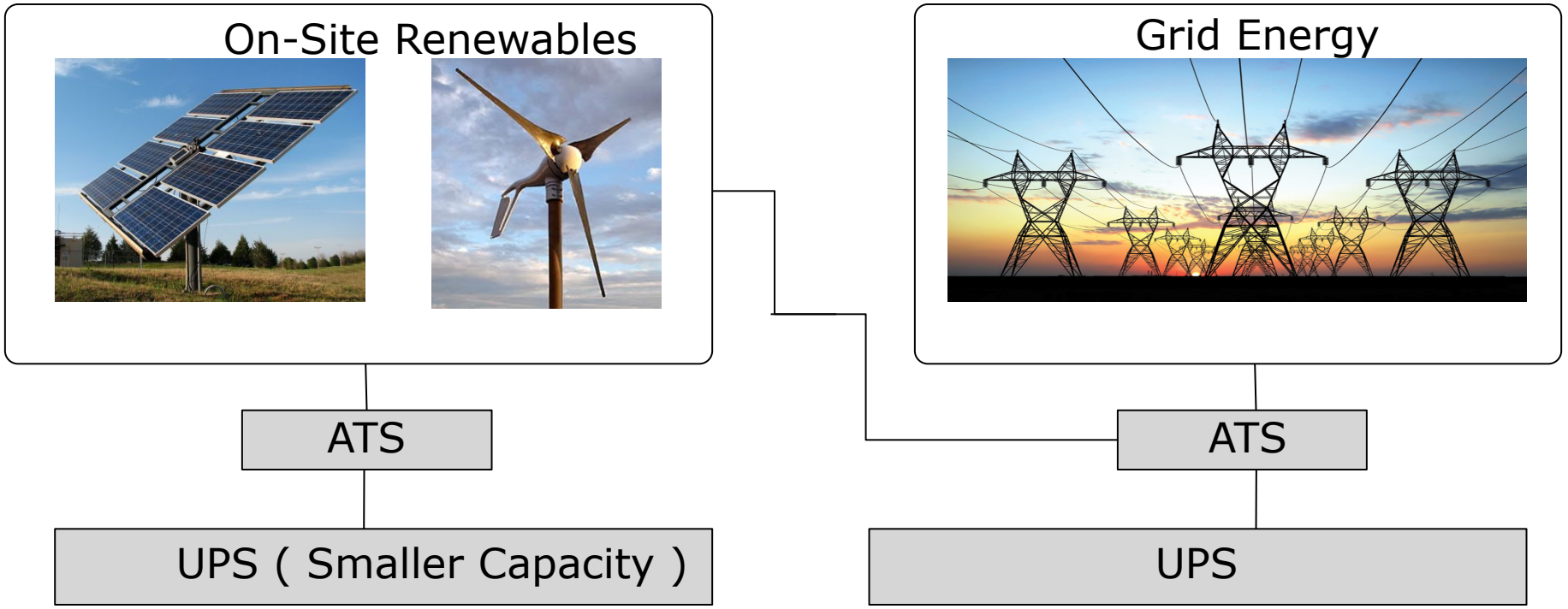
- Per-VM in-memory queues for receiving updates
  - Enables fast acknowledgements
  - Write VM memory state to disk in background
- Highly multiplexed
  - Reduces extra hardware/power required for Yank

Restoration Service



- Restoration service on stable server
  1. Receives in-memory queue+snapshot in parallel
  2. Applies in-memory queue to snapshot
  3. Restores VM using hypervisor's restoration command

Transient Servers in Green Data Centers



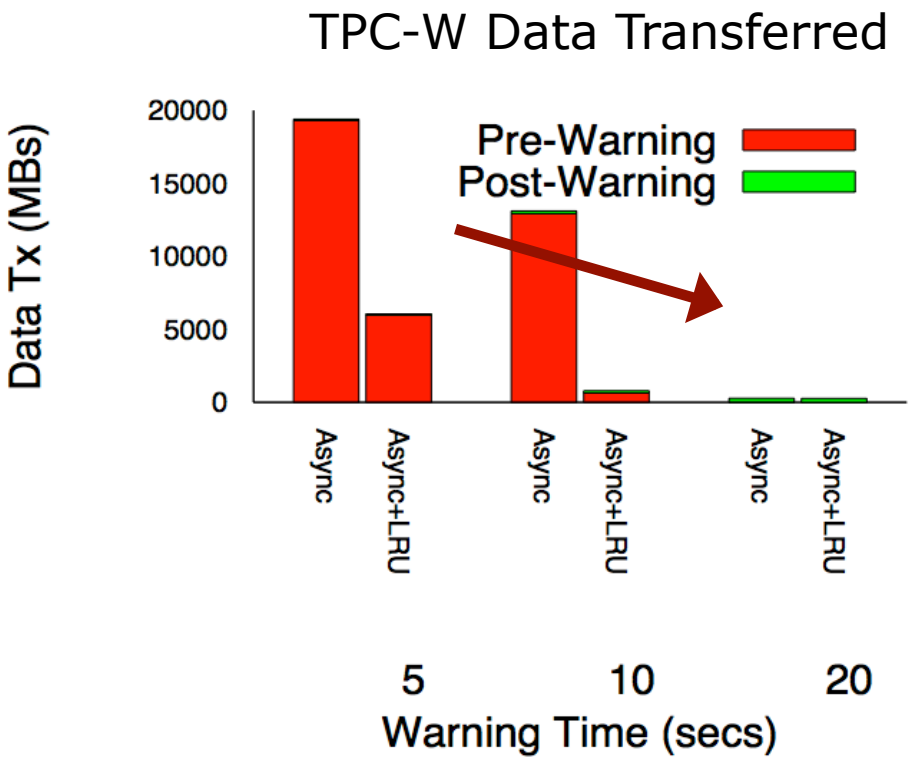
Transient Servers

Stable Servers

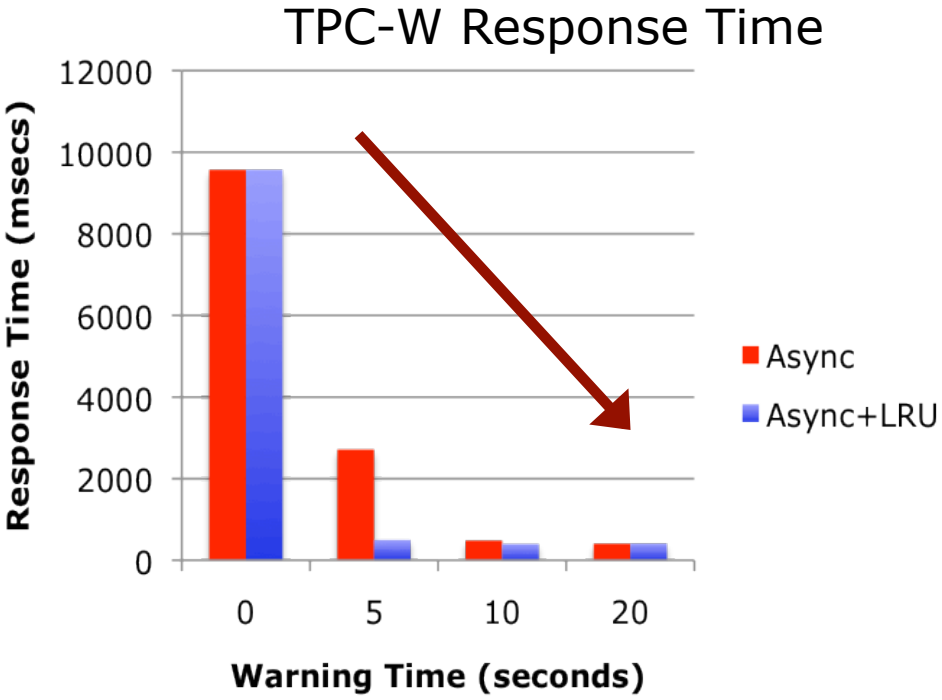
Experimental Setup

- Implementation
  - *Snapshot Manager* – modification to Remus in Xen
  - *Backup Engine* – user level Python and C code
  - *Restoration Service* – C code
- Cluster of Blade Servers
  - 4GB RAM, 2.13 GHz Processor
- Benchmark Applications
  - TPC-W: Online bookstore
- Renewable Energy generation
  - Solar/Wind Generation Traces from UMass Deployment

Exploiting Warning Time



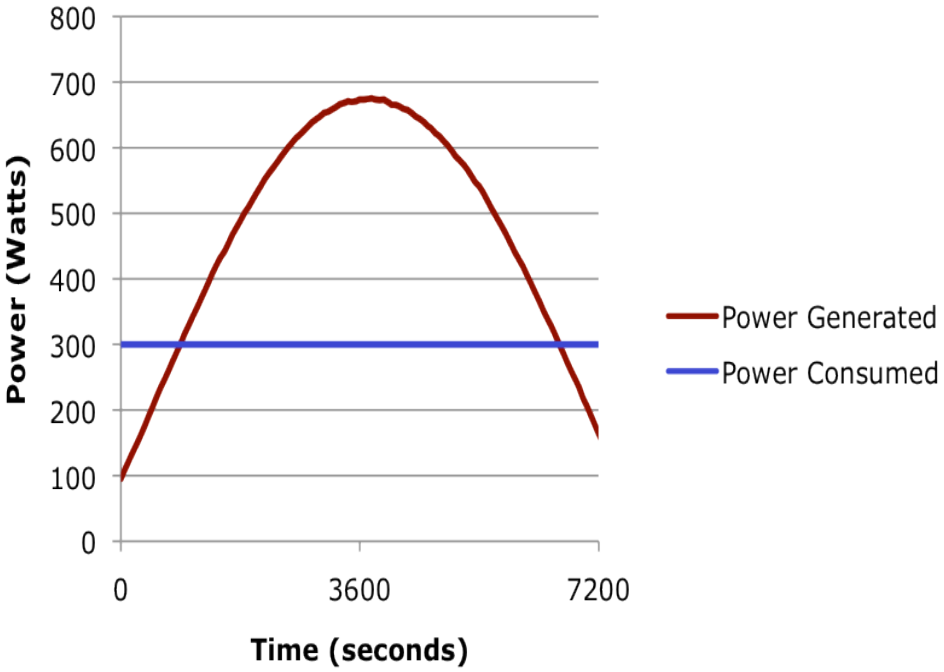
70x reduction in data transferred

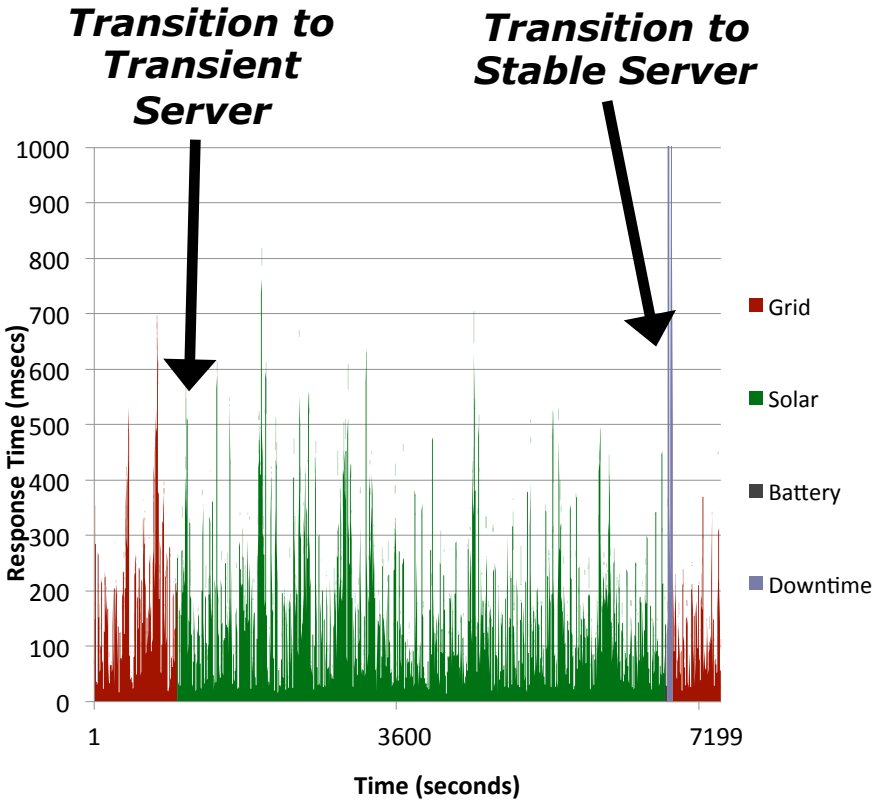


20x improvement in response time with just 5 second warning

- **4GB backup server supports 15 transient VMS**

Solar Power Driven Transiency





- **Yank masks applications from transiency due to changing power availability**

Conclusions

- Transient Servers
  - Servers that terminate after a warning
  - Applicable to many scenarios
- Yank
  - System support for transient servers
  - Virtualization-layer solution
- Evaluation
  - Low overheads – performance, hardware, power
  - Hide transiency due to renewable power
  - *Ongoing Work:* Apply to Amazon spot instances

Questions?