A Dual-System Approach to Realistic Evaluation of Large-scale Networked Systems

Richard Alimi

Thesis Defense

September 29, 2010

Committee

Y. Richard Yang (Advisor)
Michael Fischer
Sanjai Narain (Telcordia)
Avi Silberschatz

Joint work with Chen Tian, Ye Wang, Richard Yang, and David Zhang (PPLive)

Research Output

Publications

- R. Alimi, C. Tian, Y.R. Yang, D. Zhang, “PEAC: Performance Experimentation as a Capability in Production Internet Live Streaming”, *Under submission*
- L.E. Li, R. Alimi, D. Shen, H. Viswanathan, Y.R. Yang, “A General Algorithm for Interference Alignment and Cancellation in Wireless Networks”, in Infocom 2010
- Y. Wang, H. Wang, A. Mahimkar, R. Alimi, Y. Zhang, L. Qiu, Y.R. Yang, “R3: resilient routing reconfiguration”, In Sigcomm 2010
- L.E. Li, R. Alimi, R. Ramjee, H. Viswanathan, Y.R. Yang, “muNet: Harnessing Multiuser Capacity in Wireless Mesh Networks”, In Infocom 2009
- R. Alimi, L.E. Li, R. Ramjee, H. Viswanathan, Y.R. Yang, “iPack: in-Network Packet Mixing for High Throughput Wireless Mesh Networks”, In Infocom 2008
- R. Alimi, Y. Wang, Y.R. Yang, “Shadow configuration as a network management primitive”, In Sigcomm 2008
- L.E. Li, R. Alimi, R. Ramjee, J. Shi, Y. Sun, H. Viswanathan, Y.R. Yang, “Superposition coding for wireless mesh networks”, Extended abstract, In Mobicom 2007

Other Projects

- P4P: Provider Portal for Applications
- DECADE: Open Content Distribution using Data Lockers

The Official Google Blo

This is your pilot speak pattern...

5/14/2009 12:15:00 PM

Imagine if you were trying to fly from through an airport in Asia. And a bur was backed up and your journey too happened to some of our users toda

An error in one of our systems caus created a traffic jam. As a result, ab interruptions. We've been working ha especially embarrassing when a glit happened, and you can be sure that problem won't happen again. All plar

Posted by Urs Hoelzle, SVP, Opera

Skype 3.2.x -> 3.5.x upgrade problem - Connection lost

great_scandinavian

Regular member

Posts: 17

Had lots of Vista machine running just fine behind company firewall, but after "upgrading" to 3.5.x then the upgraded Skype client applications all stopped working right after the "upgrade" and the ones that I did not upgrade which now run the previous version is still running just fine. This indicate to me that it must be the 3.5.x Skype upgrade that broke my Skype client applications!!!!

If I place the upgraded machines on the DMZ side then they work just fine without changing a thing except natually to renew the DHCP lease. If I then move the same machines back on the network behind the firewall then the clients keep working, like one can make calls and chat and so forth, but after a while or if one switch user then the Skype clients go dead again.

Summary: Remember nothing else has been done other than upgrading to 3.5.x and all the older 3.2.x still work fine!

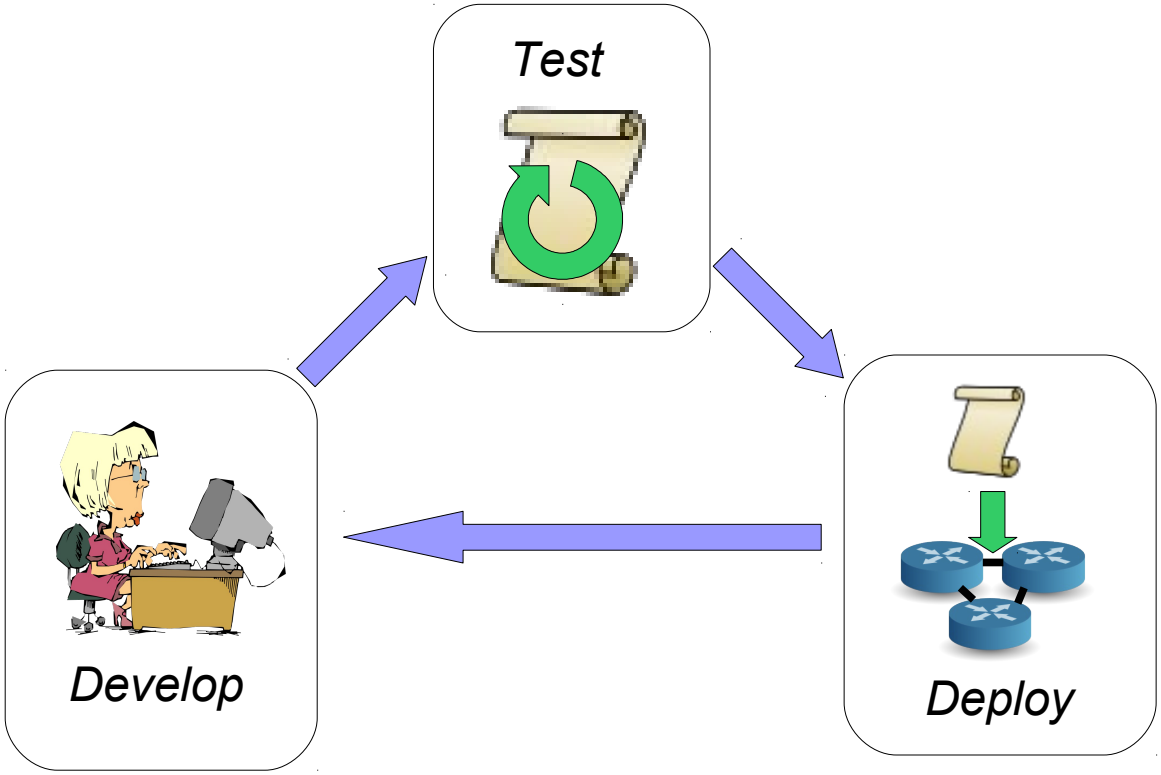
I have written to Skype bug reporting and described the issue in great detail, as it is a big problem for me being a paying customer and therefore count on the Skype services, but have not heard back from them!

Regards,

LP

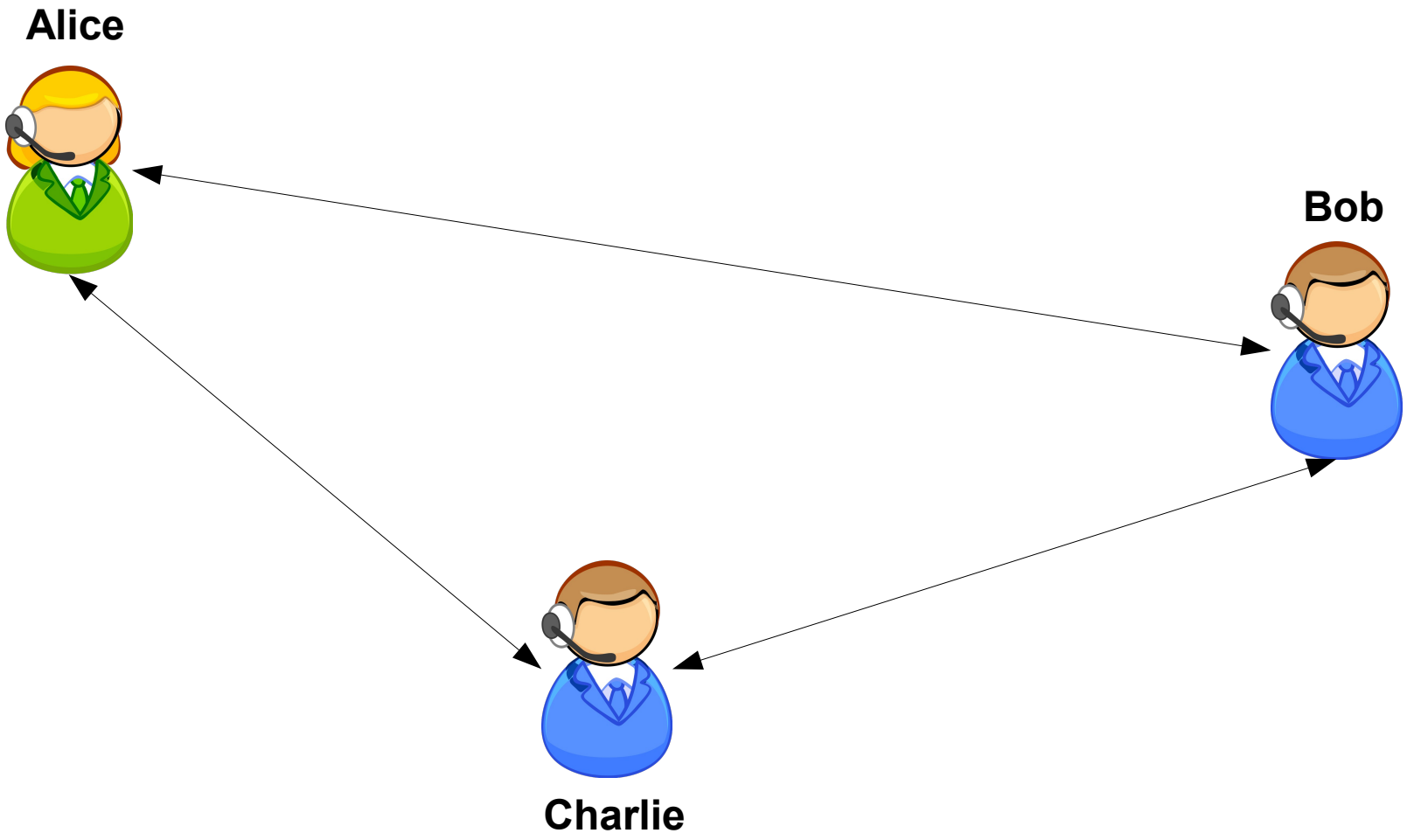
This post has been edited by **great_scandinavian**: 24 August 2007 - 06:00 AM

Development Cycle

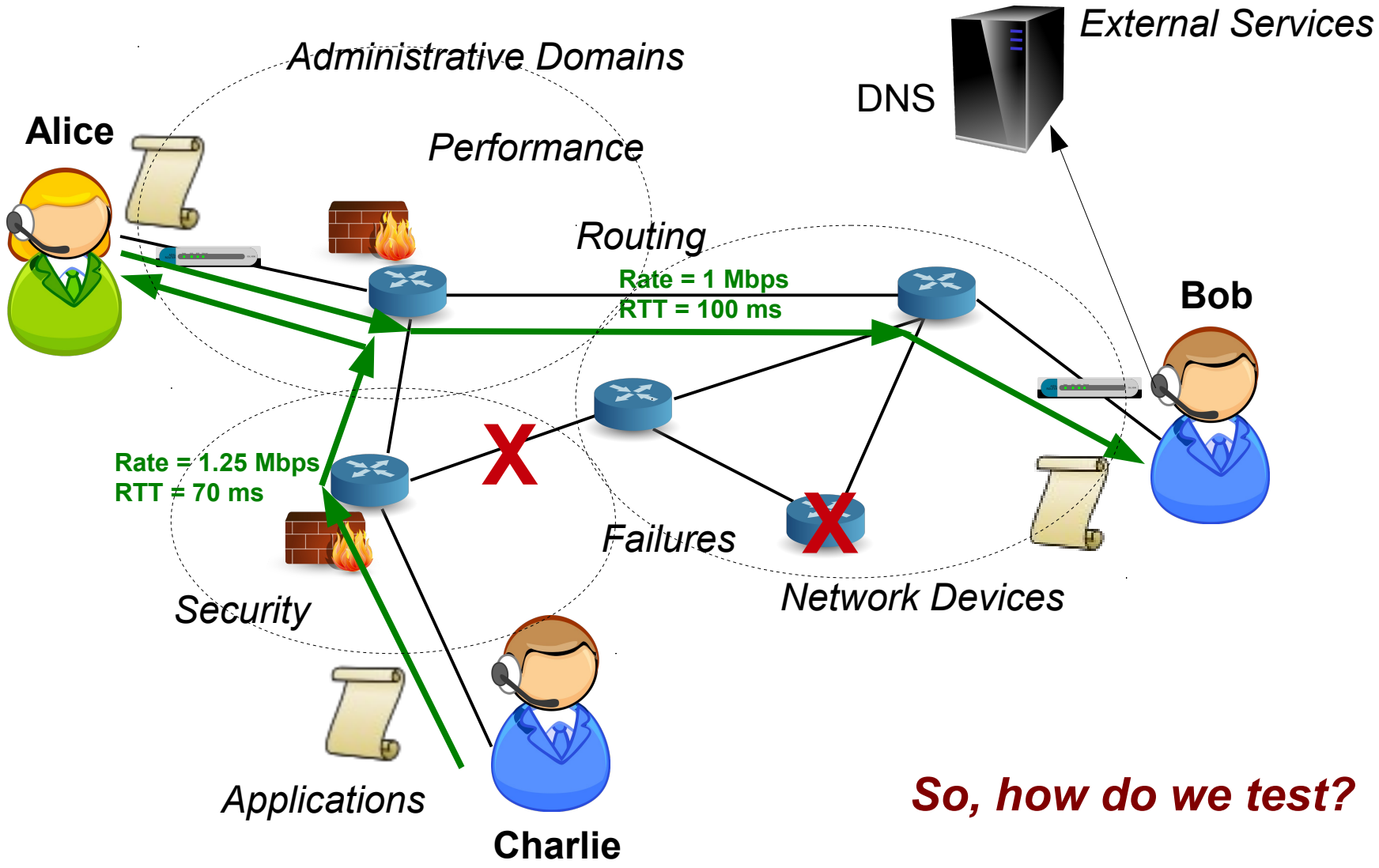


Testing is a crucial step!

Networked Systems are Simple, Right?



Networked Systems are Complex!



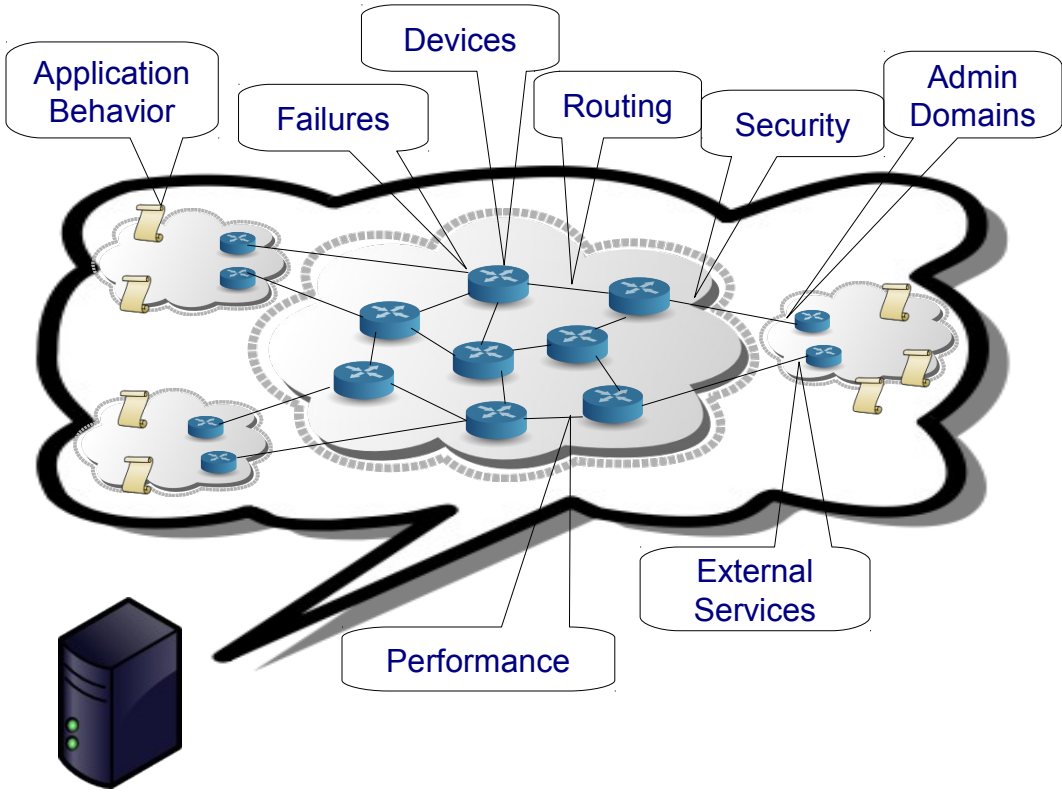
Modeling, Analysis, and Simulation

Developing a model

- Key features
- Approximations

Benefits

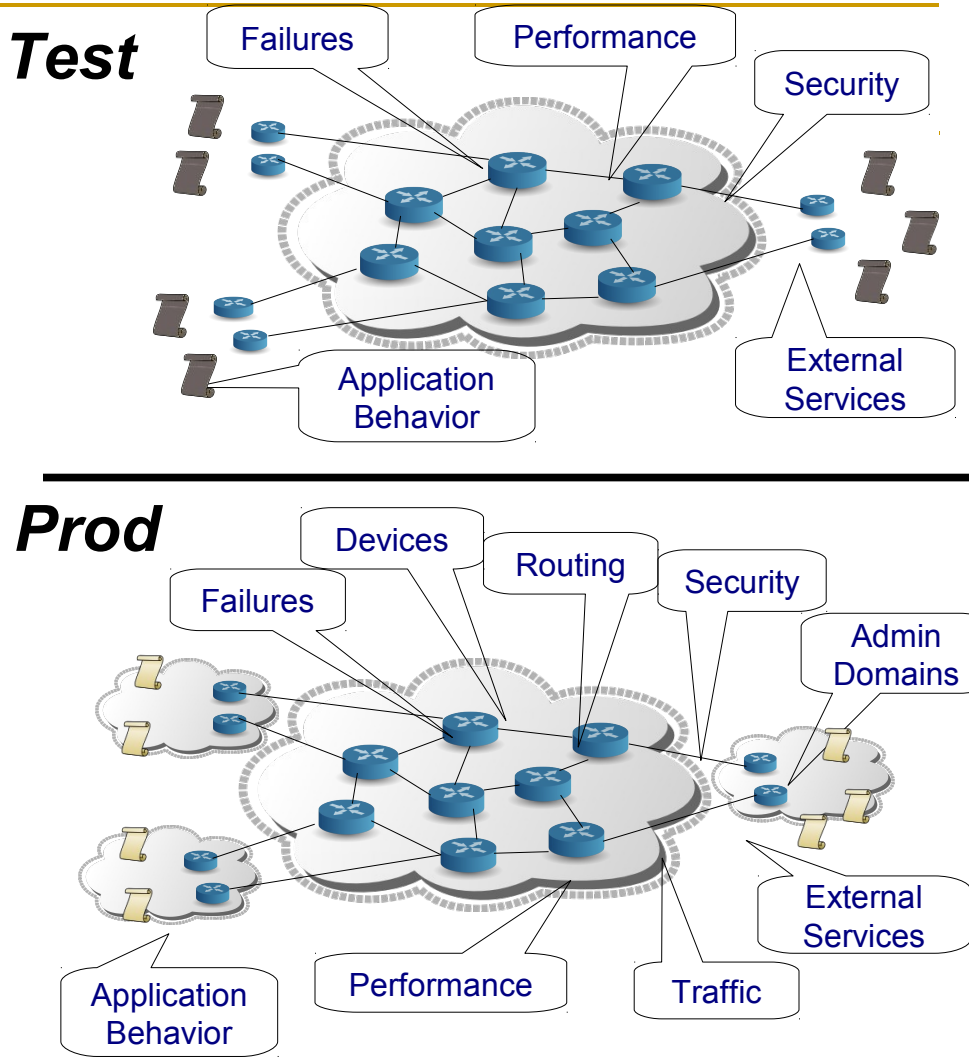
- Faster to explore impacts of changes
- Understand relationships



Limitations

- Key features may not capture all important behavior

Lab Testing



Testing infrastructure

- Separately maintained
- Similar to production infrastructure

Benefits

- Real system running
- Control test scenarios

Limitations

- Costly to maintain infrastructure similar to production
- Difficult/impossible to capture all production behaviors

Insight

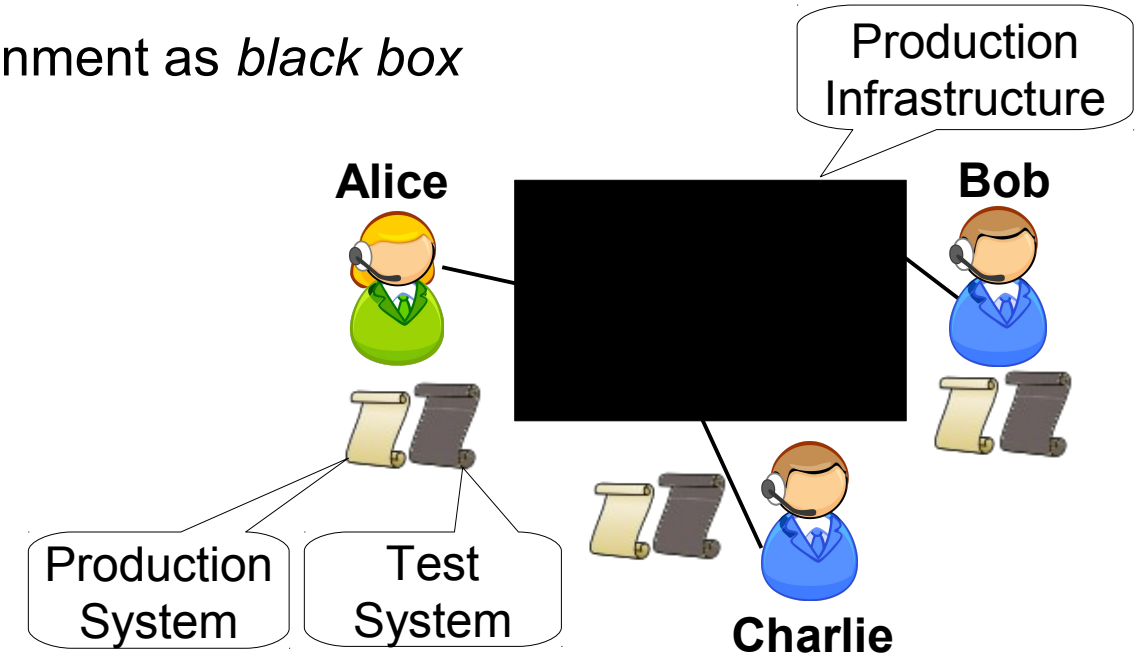
Production infrastructure meets needs for realism

- Same environment, hardware, software, etc

Run test system on production infrastructure

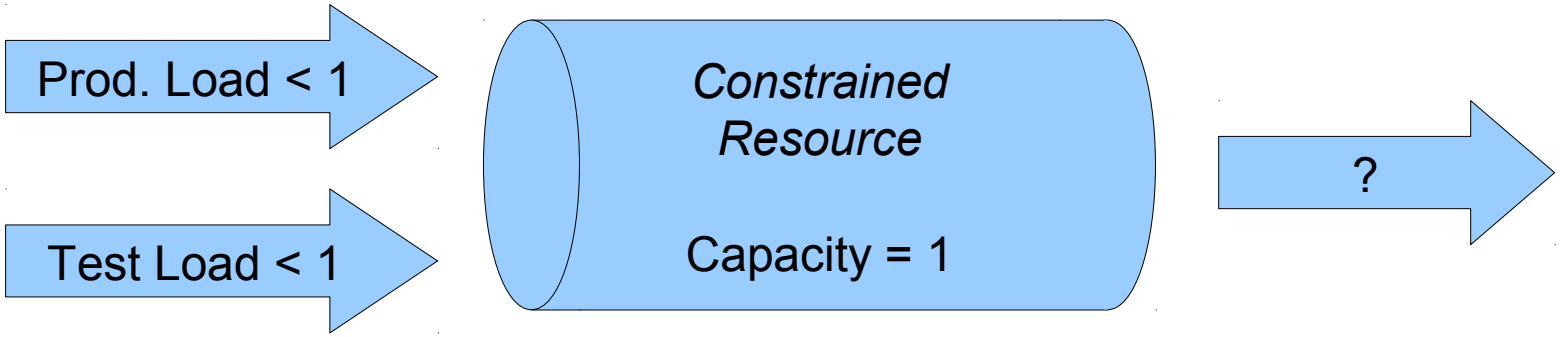
- Tests can treat environment as *black box*

But some key features are missing!



The Problem of Being Oblivious

Being oblivious to internal semantics does not suffice



Prod + Test > 1

Drop test load

- → may impact accuracy

Drop production load

- → may cause disruption to users

Insight: using domain-specific knowledge and novel techniques can resolve the conflict!

Key Questions

Performance

- *How do we avoid disruption to users?*
- *Can performance tests be accurate?*

Control

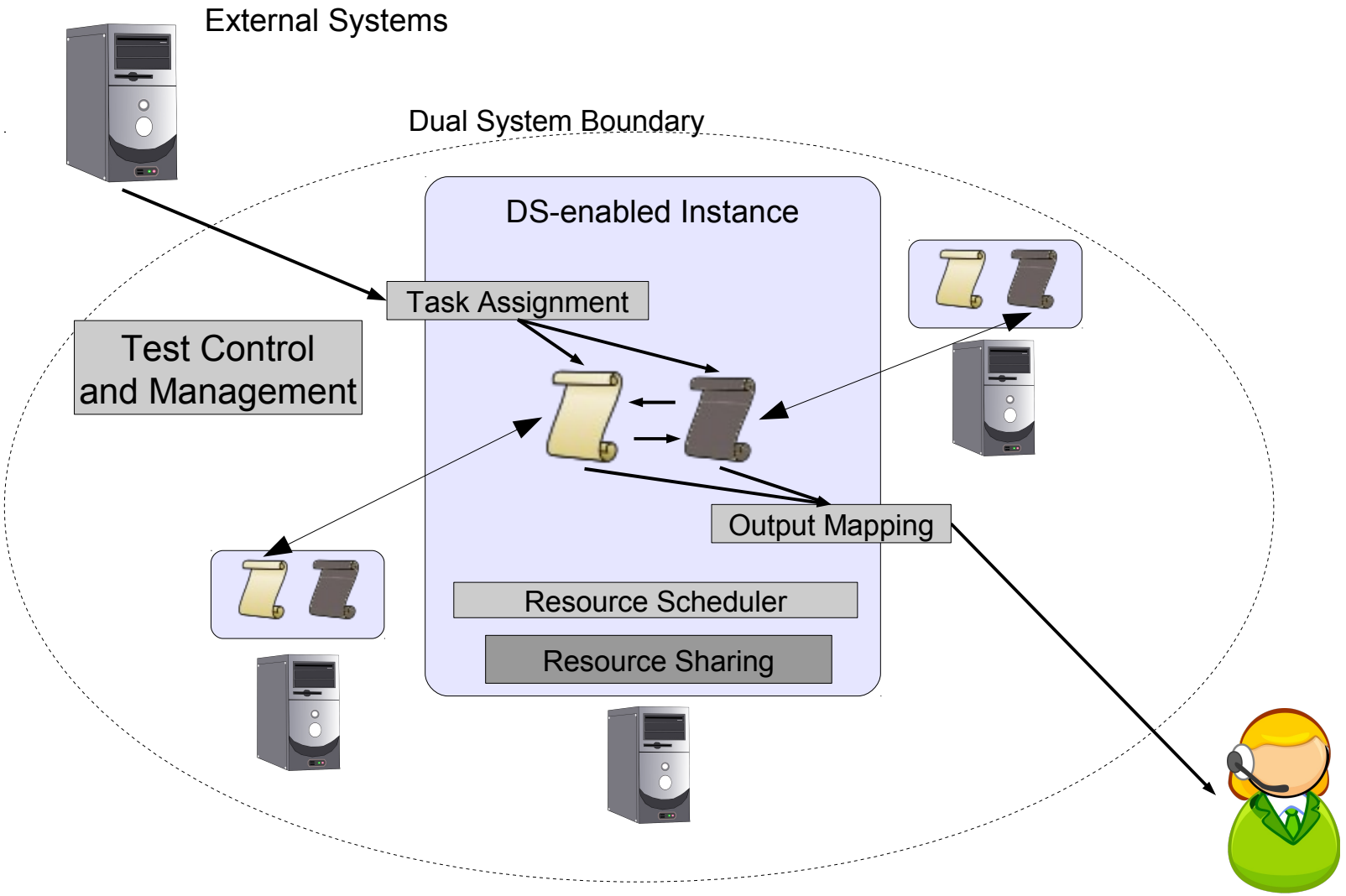
- *How can we control and manage test scenarios?*

Dual Systems

Production and test systems run *side-by-side*
on same production infrastructure

Testing and experimentation are
provided as a *basic capability*

Dual-System General Architecture



Applying Dual Systems

	PEAC P2P live streaming	***ShadowNet*** Network Configurations
Resource Sharing	Adaptive Task Reallocation	Packet Cancellation Merged FIB
Test Control	Distributed Scenario Control	Delta-debugging Shadow Traffic Control
Management	Experiment Distribution	Network-wide Commitment
Implementation Technique	Compositional Runtime	Shadow-enabled Forwarding and Control Planes

PEAC

Performance Experimentation as a Capability in Production Internet Live Streaming

Dual-system for P2P Live Streaming

R. Alimi, C. Tian, Y.R. Yang, D. Zhang, “PEAC: Performance Experimentation as a Capability in Production Internet Live Streaming”, *Under submission*

PEAC Outline

Introduction to P2P Live Streaming

PEAC Usage and Architecture

Test Control

- Distributed Scenario Control

Resource Sharing

- Adaptive Task Reallocation

Evaluations

What is P2P Live Streaming?

TorrentFreak

Home

CNN Uses P2P Plugin for its Live Stream

Written by Ernesto on January 24, 2009

This week, millions of people watched Obama's inauguration on the Internet through one of the many sites that offered a live feed. CNN's broadcast was without doubt one of the most used viewed streams, with a peak of more than a million simultaneous viewers and also one that was using P2P technology.

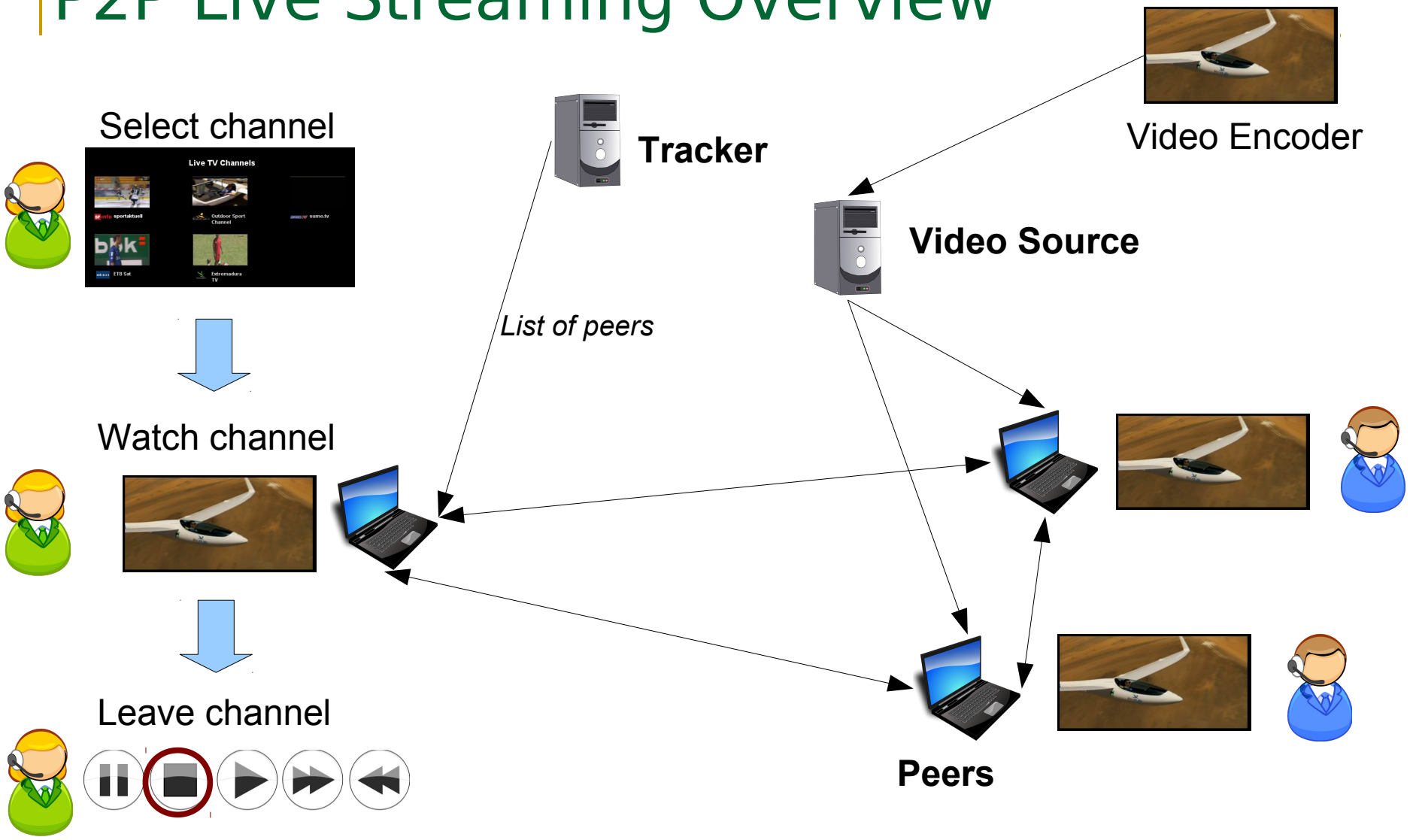
PPLive

Used to deliver both major events...

… and daily viewing

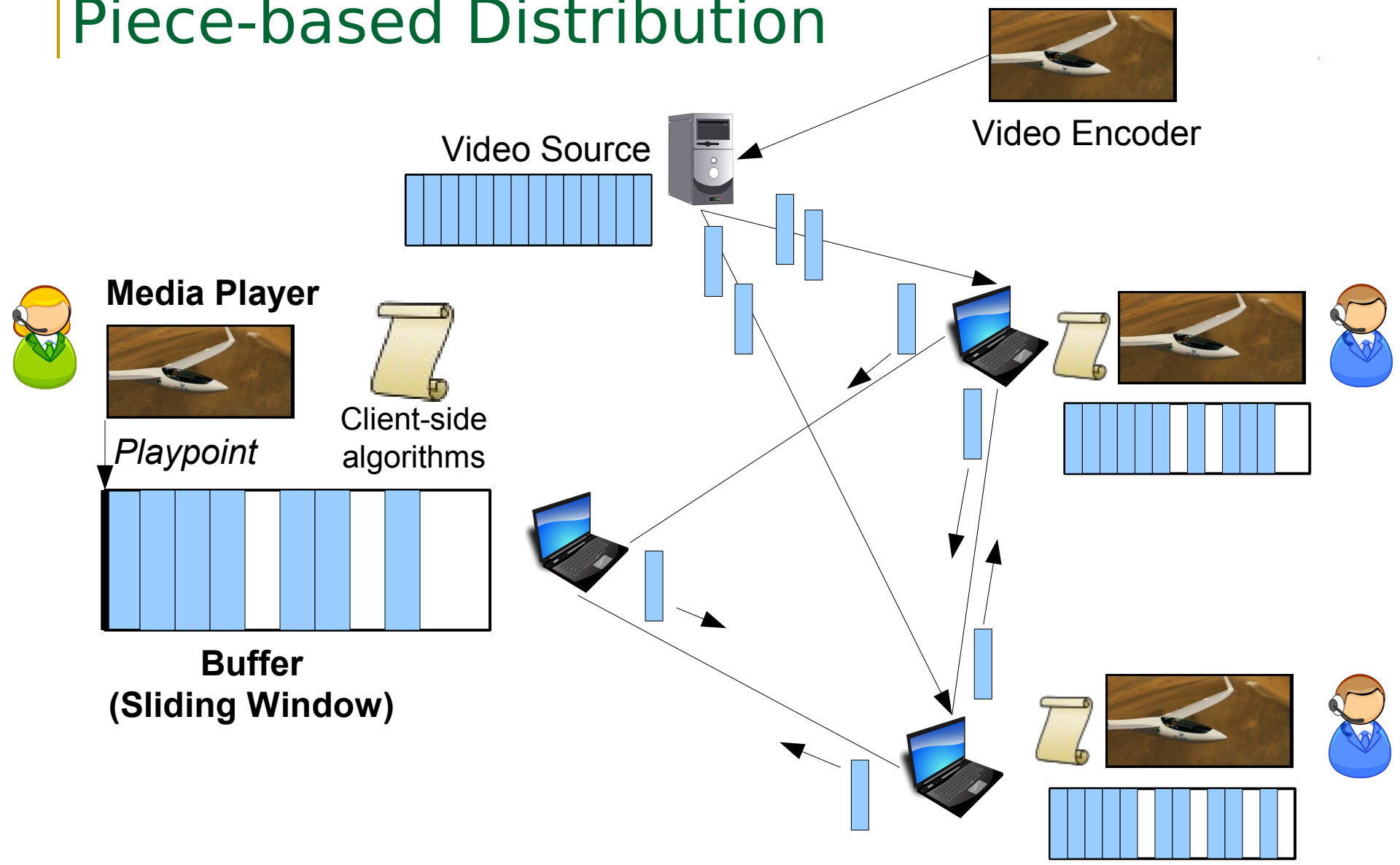
Expanding set of applications now include P2P support (e.g., Adobe Flash 10.1)

P2P Live Streaming Overview



Screenshot image source: http://www.zattoo.com

Piece-based Distribution



Algorithmic Components

Topology management

- From *whom* do I download?

Piece selection

- *What* do I download?

Rate control

- *How much* do I download?

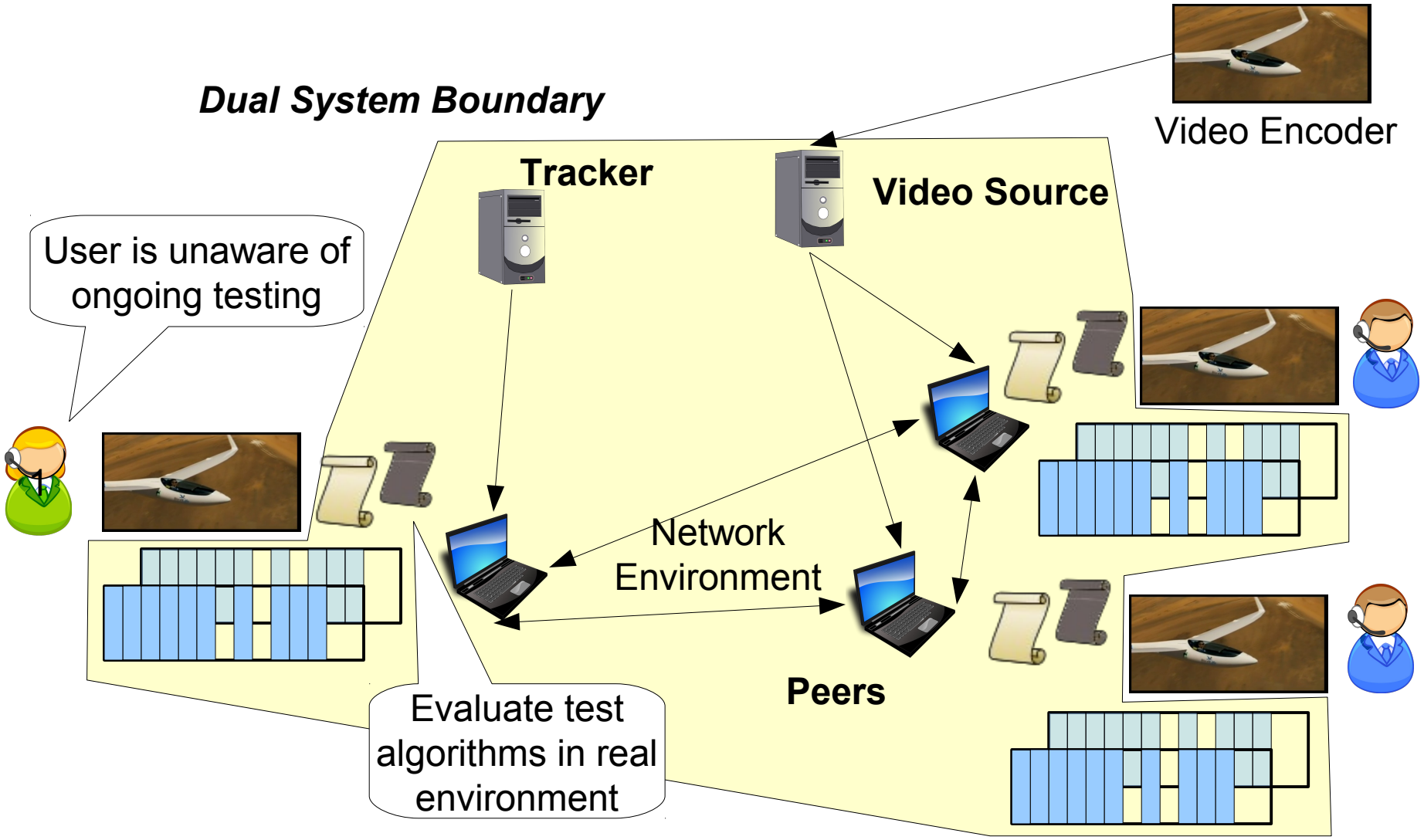
Scenario-specific algorithms

- Coordinated usage of shared bottleneck (e.g., enterprise)
- Flash-crowd admission control
- Use network information
- Use in-network storage
- ...

Dual system architecture lets us test algorithms and network environment as black boxes!

All of these can affect video quality → testing is crucial!

Dual System for P2P Live streaming



PEAC Outline

Introduction to P2P Live Streaming

PEAC Usage and Architecture

Test Control

- Distributed Scenario Control

Resource Sharing

- Adaptive Task Reallocation

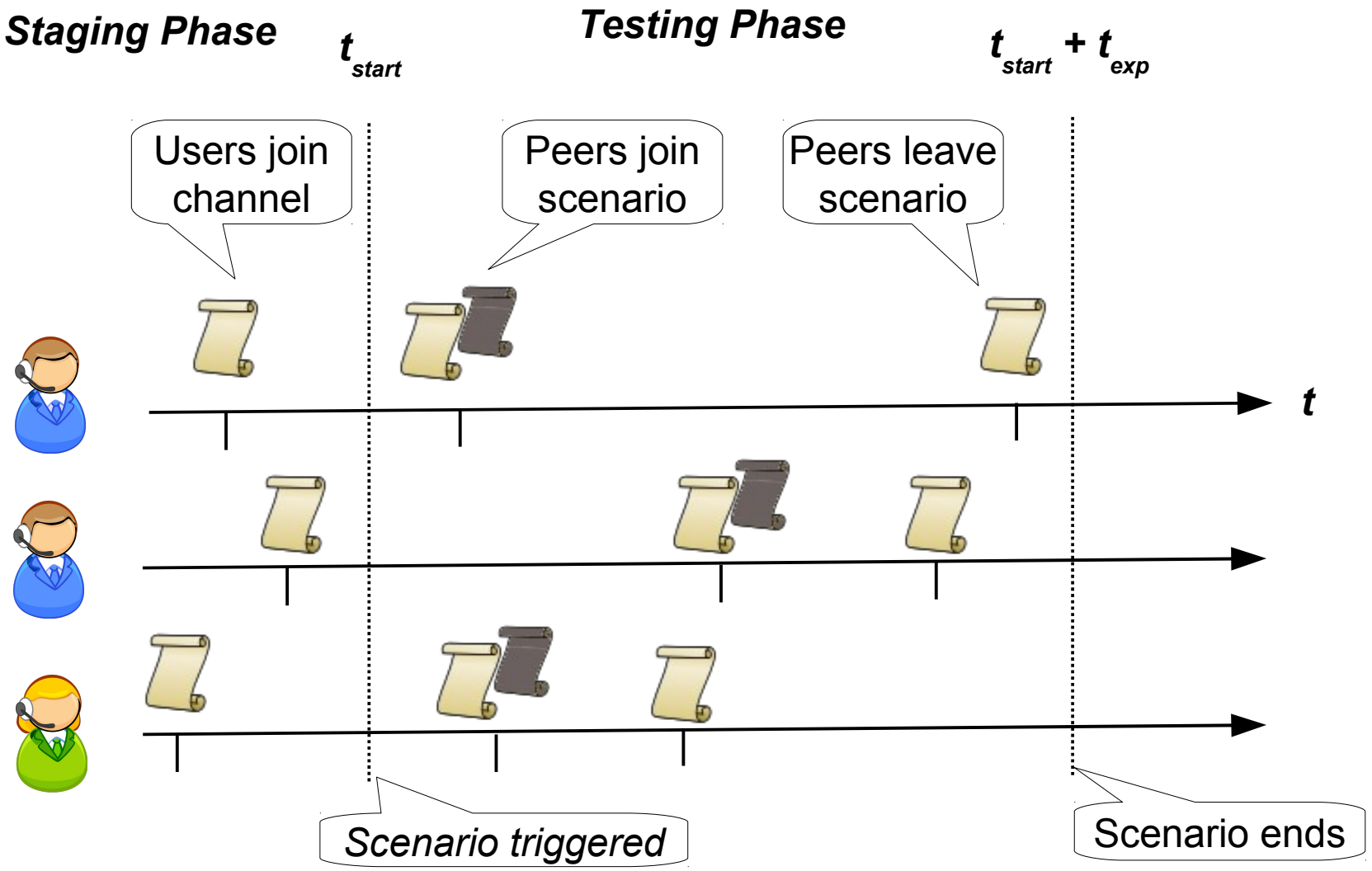
Evaluations

PEAC Usage

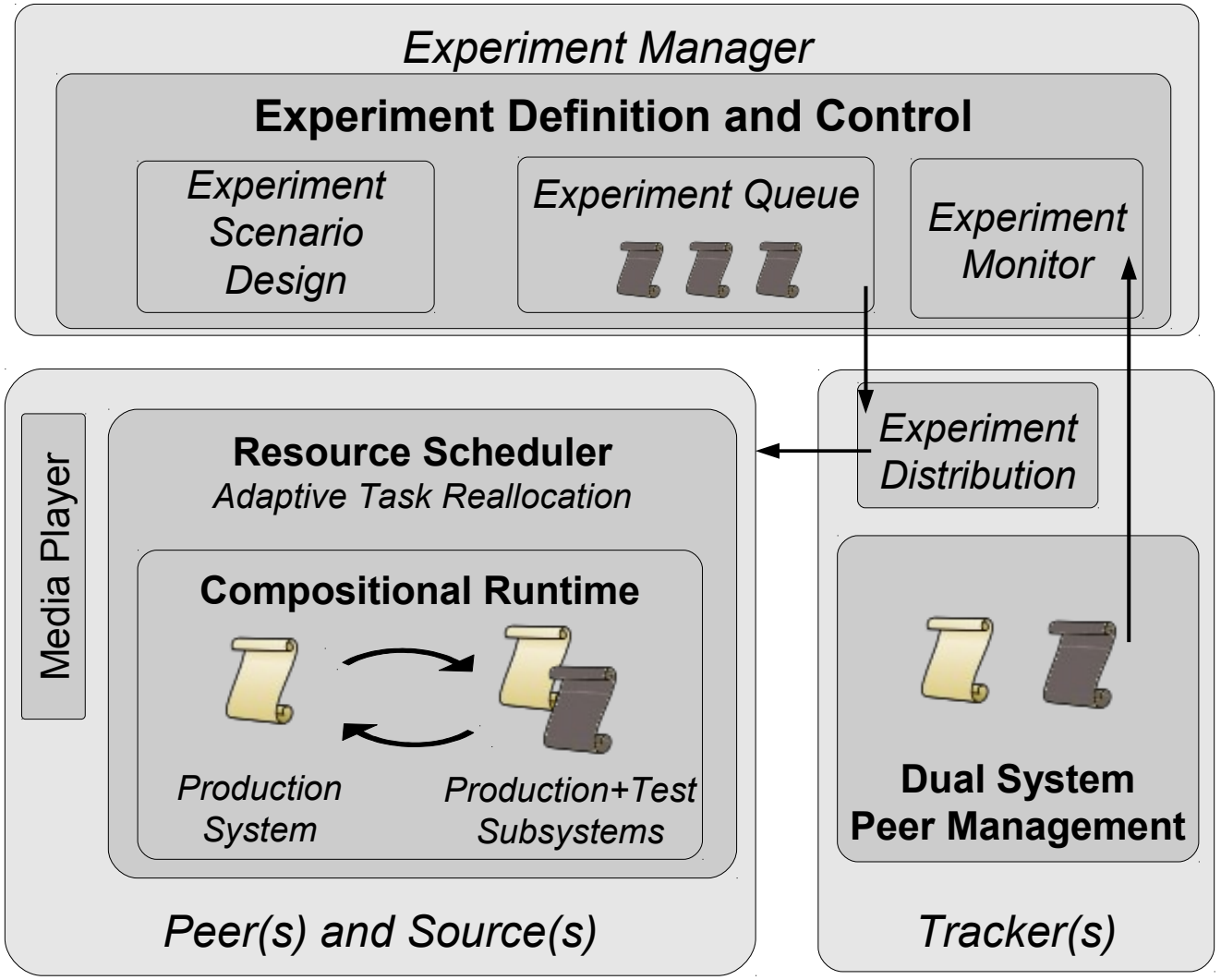
Basic usage

- Set of channels are available on production infrastructure
- Developer defines *experiments*
  - Each *experiment* consists of *scenarios* executed in parallel
  - *Scenario* defines set of parameters for a test
    - Consists of *peer behavior configuration* and *algorithms*
    - Performance measurements dependent on both
- PEAC monitors channels for feasibility and executes experiments

Executing a Test Scenario



PEAC Architecture



PEAC Outline

Introduction to P2P Live Streaming

PEAC Usage and Architecture

Test Control

- Distributed Scenario Control

Resource Sharing

- Adaptive Task Reallocation

Evaluations

Experiment Definition and Control

A scenario's *peer behavior configuration* is defined by

- Peers selected to run the test system
  - May select based on peer properties (estimated capacity, location, etc)
- Arrival behavior
  - Arrival rate may vary with time
- Peer lifetime
  - Developer indicates desired peer lifetimes
- User behavior in relation to viewing quality
  - Developer defines conditions (e.g., freezes) under which peers depart

Peer Arrival Problem Definition

Given

- Experiment start time t_{start}
- Time-varying arrival rate $\lambda(t)$ on $[0, t_{exp}]$
  - Flexibility to create flash-crowds, “steady-state” scenarios, etc

Devise algorithm such that each peer i computes arrival time a_i given $\lambda(t)$, t_{start}, t_{exp}

Distributed Scenario Control (DSC)

Straightforward solution: centralized control

- More difficult to scale to large number of peers
- Message delivery from controller may be difficult (e.g., NATs)

Distributed Control

- Tracker broadcasts scenario *parameters* to peers
  - May be distributed via P2P overlay, tracker keepalive, CDN
  - Lightweight and simple
- Each peer *locally* determines (without coordination) its arrival time

- → *Decouple* scenario *definition* from its *execution*
- → Soft-state at tracker eases scalability and reduces complexity

DSC: Peer Arrivals

Theorem

- Given $\lambda(t)$, compute expected arrivals over duration t_{exp} (denote as m)
- Choose n from Poisson distribution with mean m
- Independently draw n arrival times from a particular distribution
- Result is Poisson process with rate $\lambda(t)$

Theorem 2 (Cox and Lewis, 1962). *Let $T_1, T_2, \ldots$ be random variables representing the event times of a nonhomogeneous Poisson process with continuous expectation function $\Lambda(t)$, and let N_t represent the total number of events occurring before time t in the process. Then, conditional on the number of events $N_{t_0} = n$, the event times $T_1, T_2, \ldots, T_n$ are distributed as order statistics from a sample with distribution function $F(t) = \Lambda(t)/\Lambda(t_0)$ for $t \in [0, t_0]$.*

Source: http://filebox.vt.edu/users/pasupath/papers/nonhompoisson_streams.pdf

- Can we make this work?

DSC: Peer Arrivals – Exact Solution

Tracker:

01. Generate n from $N_{t_{\text{exp}}} \sim \text{Poisson}(\Lambda(t_{\text{exp}}))$
02. Send t_{start}, t_{exp}, and $\lambda(t)$ to n chosen peers

Peer i, upon receiving t_{start}, t_{exp}, and $\lambda(t)$:

03. Draw waiting time w_i according to $F(t) = \frac{\Lambda(t)}{\Lambda(t_{\text{exp}})}$
04. Compute arrival time: $a_{e,i} = t_{\text{start}} + w_i$

Problem

- How do we send to *n* chosen peers?
  - Requires hard-state at tracker

DSC: Peer Arrivals – Approx. Solution

Approximate solution for choosing *n* peers (out of total *M*)

- Choose *n* according to $\hat{N}_{t_{\text{exp}}} \sim \text{Binomial}(M, \frac{\Lambda(t_{\text{exp}})}{M})$

Tracker:

01. Let M be the total number of available peers
02. Let $p = \frac{\Lambda(t_{\text{exp}})}{M}$
03. Send t_{start}, t_{exp}, $\lambda(t)$, and p to each peer

Peer i, upon receiving t_{start}, t_{exp}, $\lambda(t)$, and p:

04. **if** $random() > p$ **then return**
05. Draw waiting time w_i according to $F(t) = \frac{\Lambda(t)}{\Lambda(t_{\text{exp}})}$
06. Compute arrival time: $a_{e,i} = t_{\text{start}} + w_i$

- Benefits
  - Simple, soft-state implementation for tracker

DSC: Handling Failures

User-initiated departures

- Use replacement peer

PEAC Outline

Introduction to P2P Live Streaming

PEAC Usage and Architecture

Test Control

- Distributed Scenario Control

Resource Sharing

- Adaptive Task Reallocation

Evaluations

Resource Scheduler Requirements

- Production and test systems are responsible for completing *tasks*
  - *Task* is a piece that needs to be downloaded

A_{prod} Tasks and resources assigned to *Production* running alone

A_{test} Tasks and resources assigned to *Test* running alone

A_{dual} Tasks and resources assigned to *Dual System*

Two requirements

- R1: Disruption protection $Perf(A_{dual}) >= Perf(A_{prod})$
- R2: Experimental accuracy **obtain** $Perf(A_{test})$ **from** $Perf(A_{dual})$

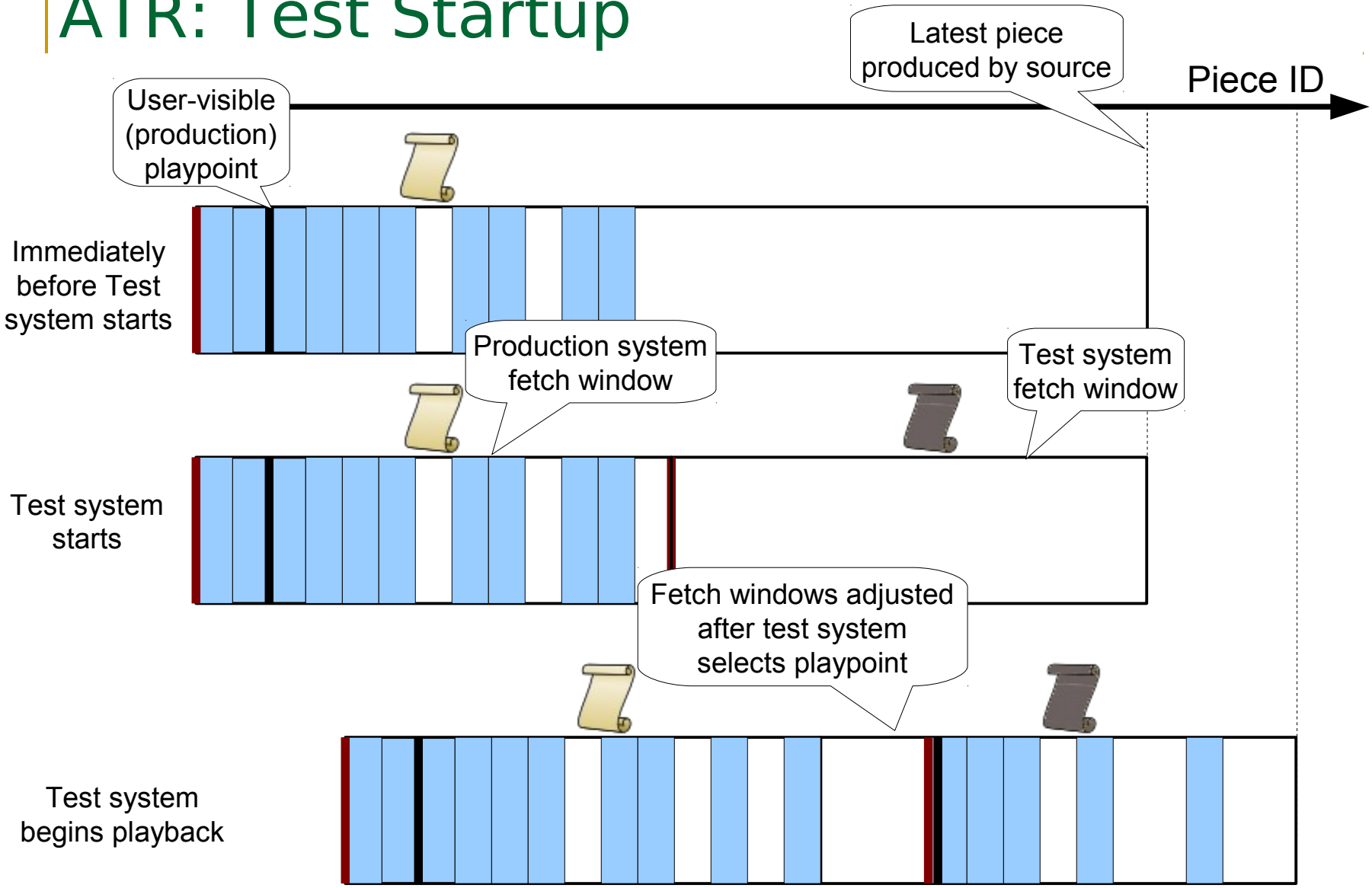
Adaptive Task Reallocation (ATR)

Basic Idea

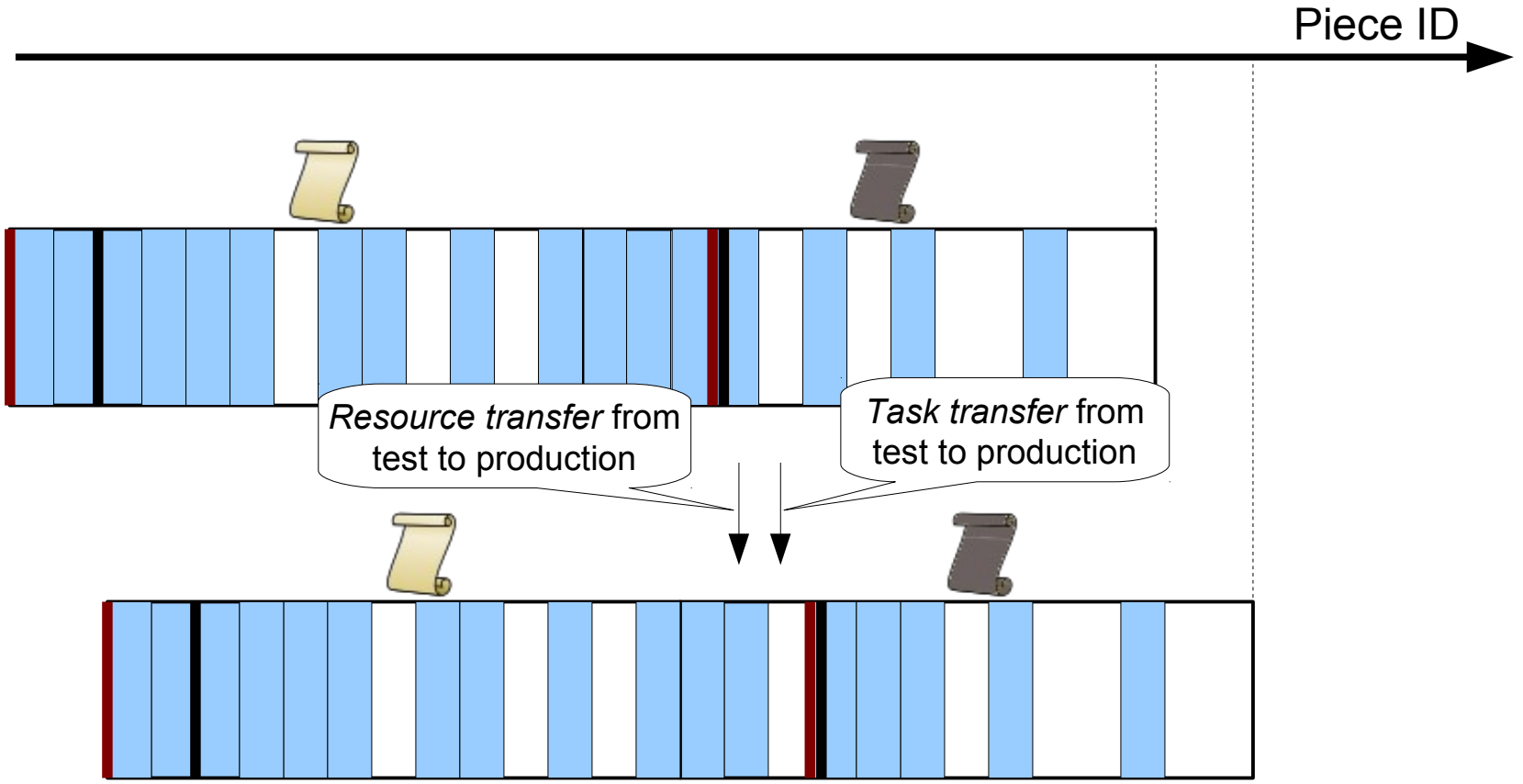
- Try to give test system the same:
  - tasks,
  - resources,
  - lag from source,
  - deadlines, and
  - block availability

  as if running alone
- When Test misses a piece's deadline, task shifted to Production
- Production given some time ($T_{recover}$) to recover missed pieces
  - User playpoint has lag compared to test system's playpoint

ATR: Test Startup



ATR: Steady State



ATR: Analysis

Data Flow Constraints

- Test → Production
- ~~Production → Test~~

Accuracy

- High accuracy when test system performs well
- Measured performance is lower bound if protection triggered
  - Due to resource competition

Overhead

- Additional lag from source may not be tolerable in all cases

PEAC Outline

Introduction to P2P Live Streaming

PEAC Usage and Architecture

Test Control

- Distributed Scenario Control

Resource Sharing

- Adaptive Task Reallocation

Evaluations

PEAC Evaluations

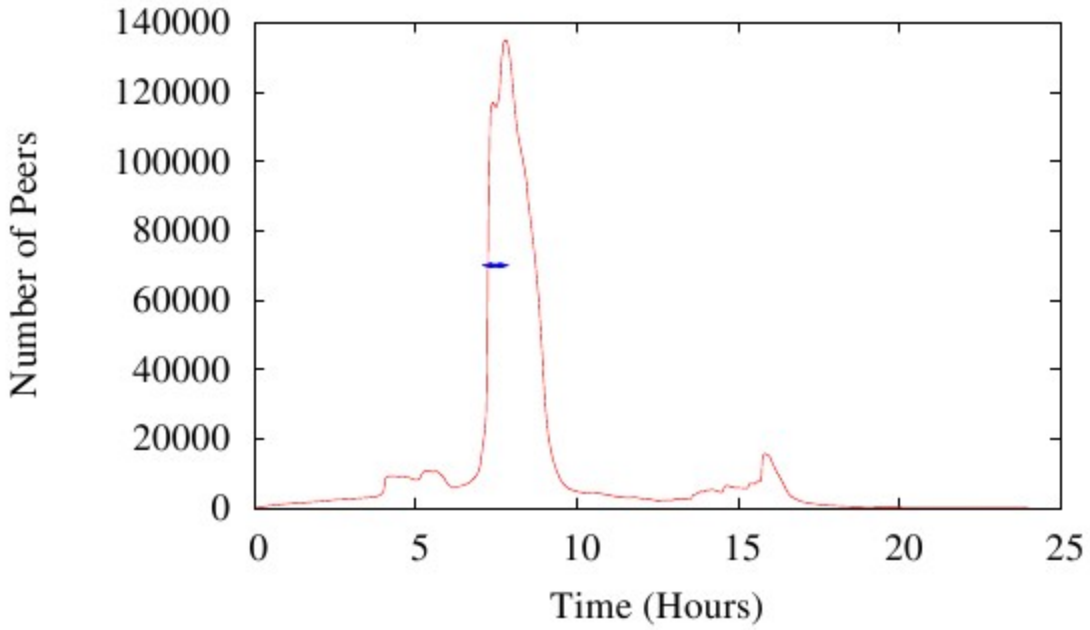
Fully implemented

- Clients, trackers, video source/encoder, and experiment manager

Evaluation methodology

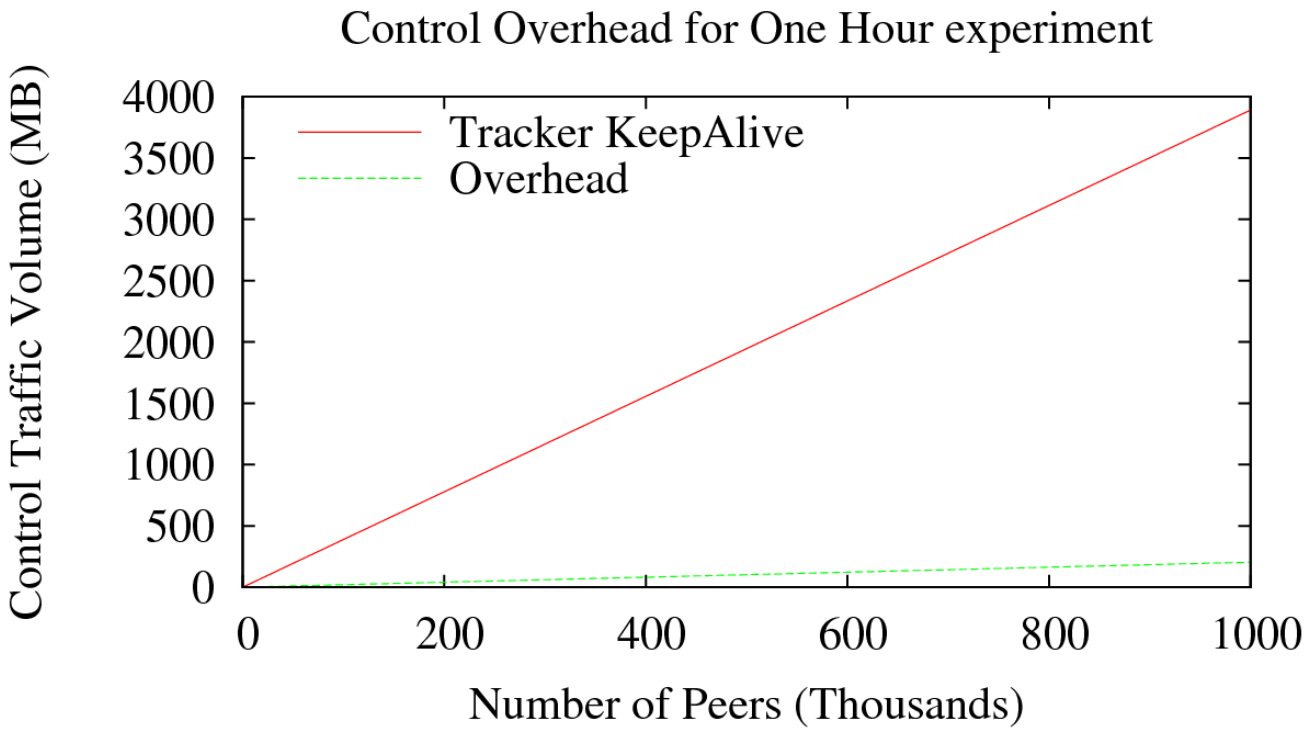
- Distributed Scenario Control
  - Emulation used to achieve large-scale
  - Driven by
    - PPLive full-day channel traces: SH Sports and HN Satellite
    - 4-hour baseball game broadcast (peak of about 60,000 peers)
- Adaptive Task Reallocation
  - Directly use implementation
  - Run using Emulab/Modelnet with 120 clients

Test Triggering Opportunities



Opportunities to trigger 1-hour, 70,000-peer experiment in PPLive's SH Sports channel

Scenario Distribution Overhead



Traffic volume for distributing scenario parameters for a 1-hour experiment

Accuracy and User-visible Performance

Test Accuracy

	Buggy	Production using CDN	Production using P2P
Pieces Missed	4.37%	4.37%	4.48%

User-visible Performance

	Production using CDN	Production using P2P
Pieces Missed	0.0%	0.04%

Run “buggy” algorithm in test system

- Disconnects inactive peers after 1 second (instead of 5 seconds)

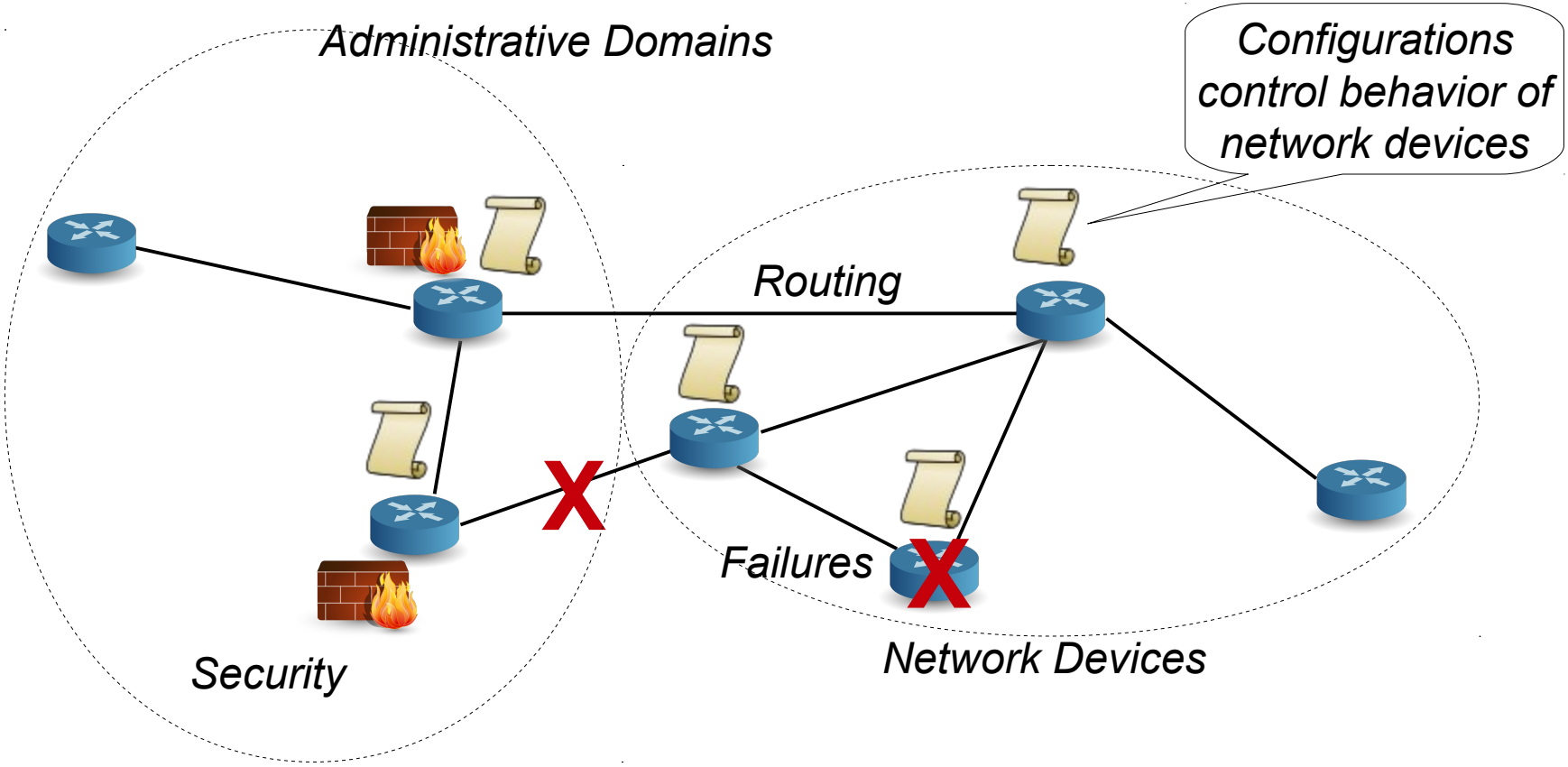
ShadowNet

Shadow Configurations as a Network Management Primitive

Dual-system for Network Configuration

R. Alimi, Y. Wang, Y.R. Yang, “Shadow configuration as a network management primitive”, In Sigcomm 2008

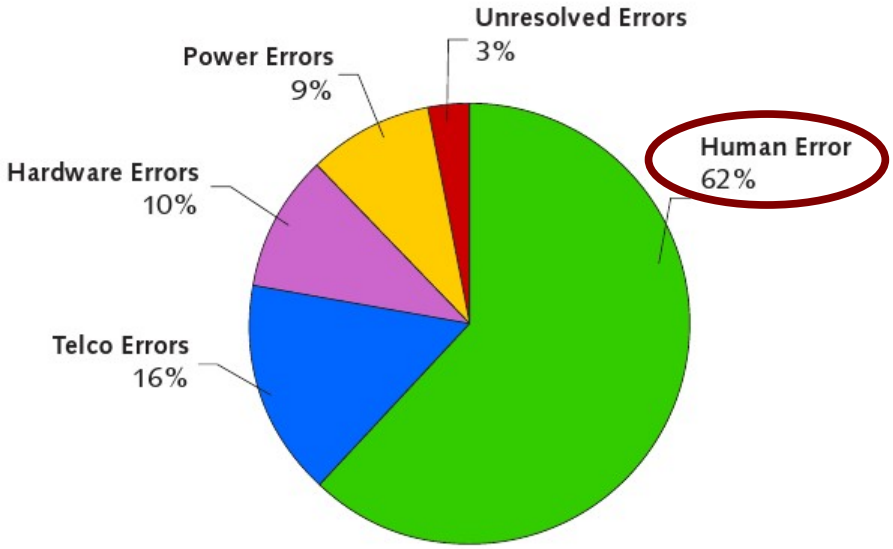
What are Network Configurations?



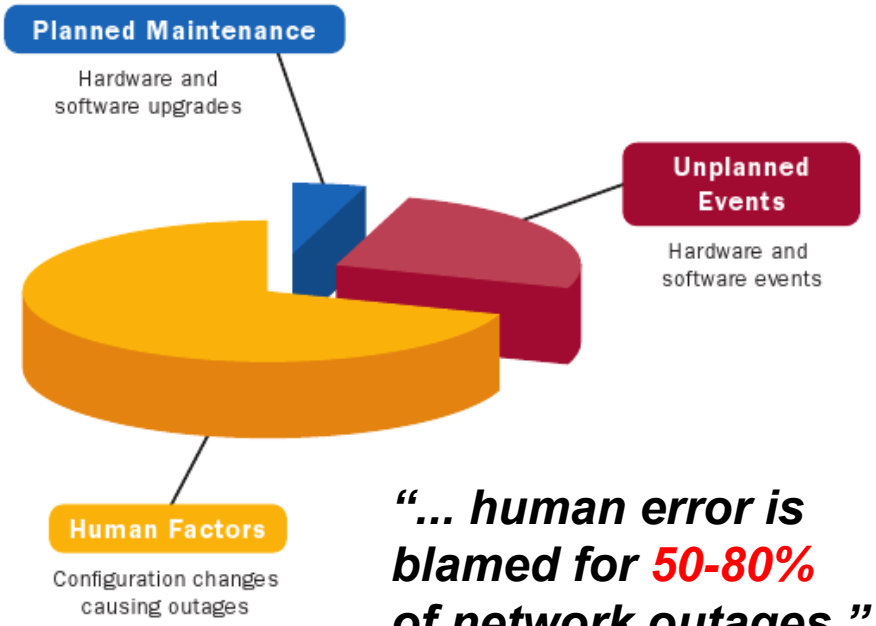
Configurations control many complex and interacting services

Configuration Leads to Errors

"80% of IT budgets is used to maintain the status quo."



Source: The Yankee Group, 2004



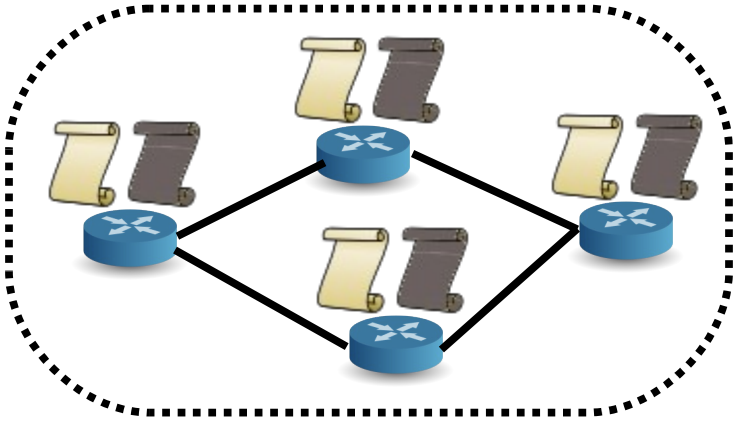
"... human error is blamed for 50-80% of network outages."

Source: Juniper Networks, 2008

ShadowNet

Key ideas

- Allow test (shadow) config on each router in addition to production config
- In-network, interactive testing environment
- “Shadow” term from computer graphics



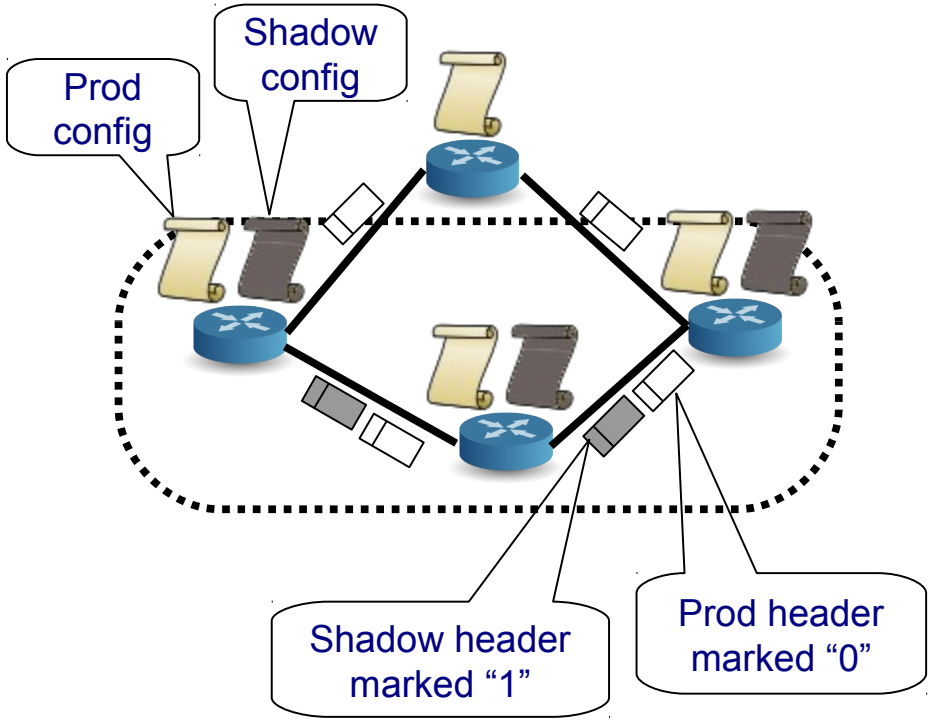
Key Benefits

- Realistic (no model)
- Scalable
- Access to real traffic
- Transactional

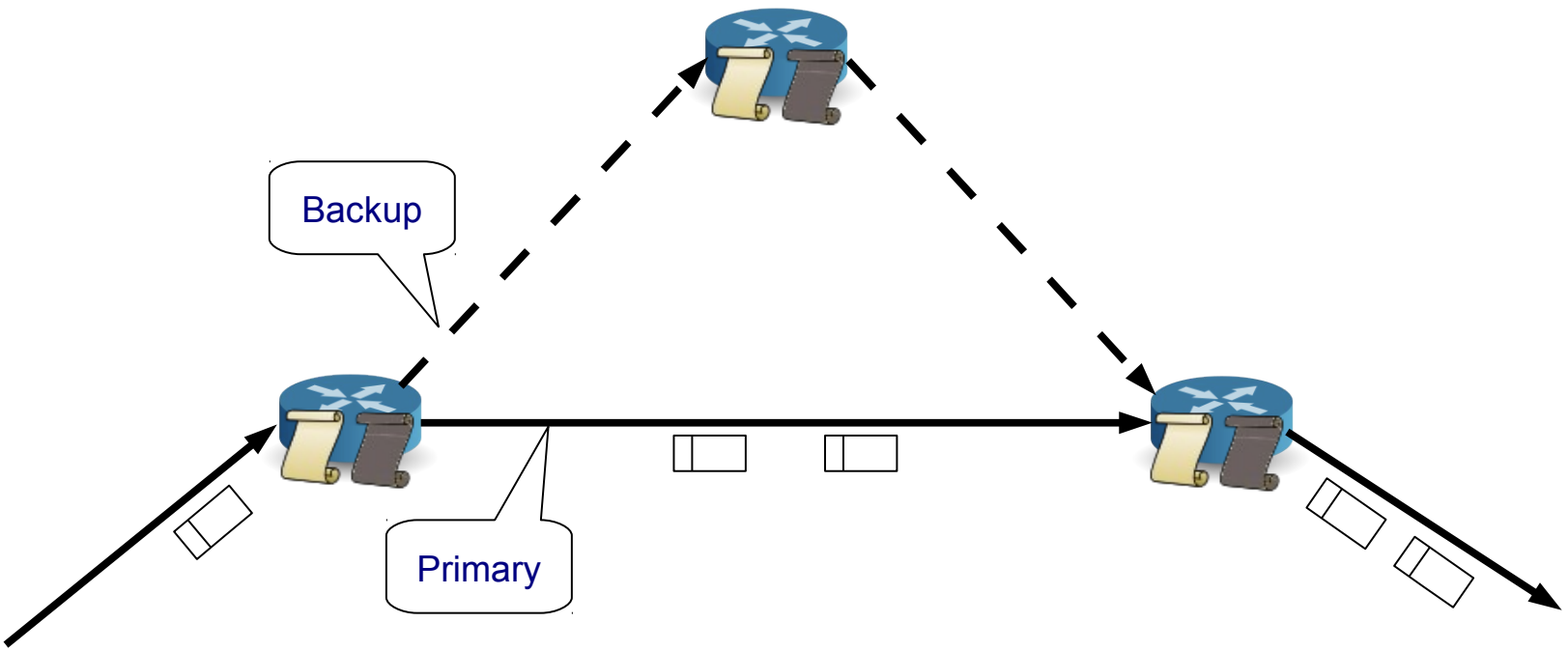
System Basics

What's in the shadow configuration?

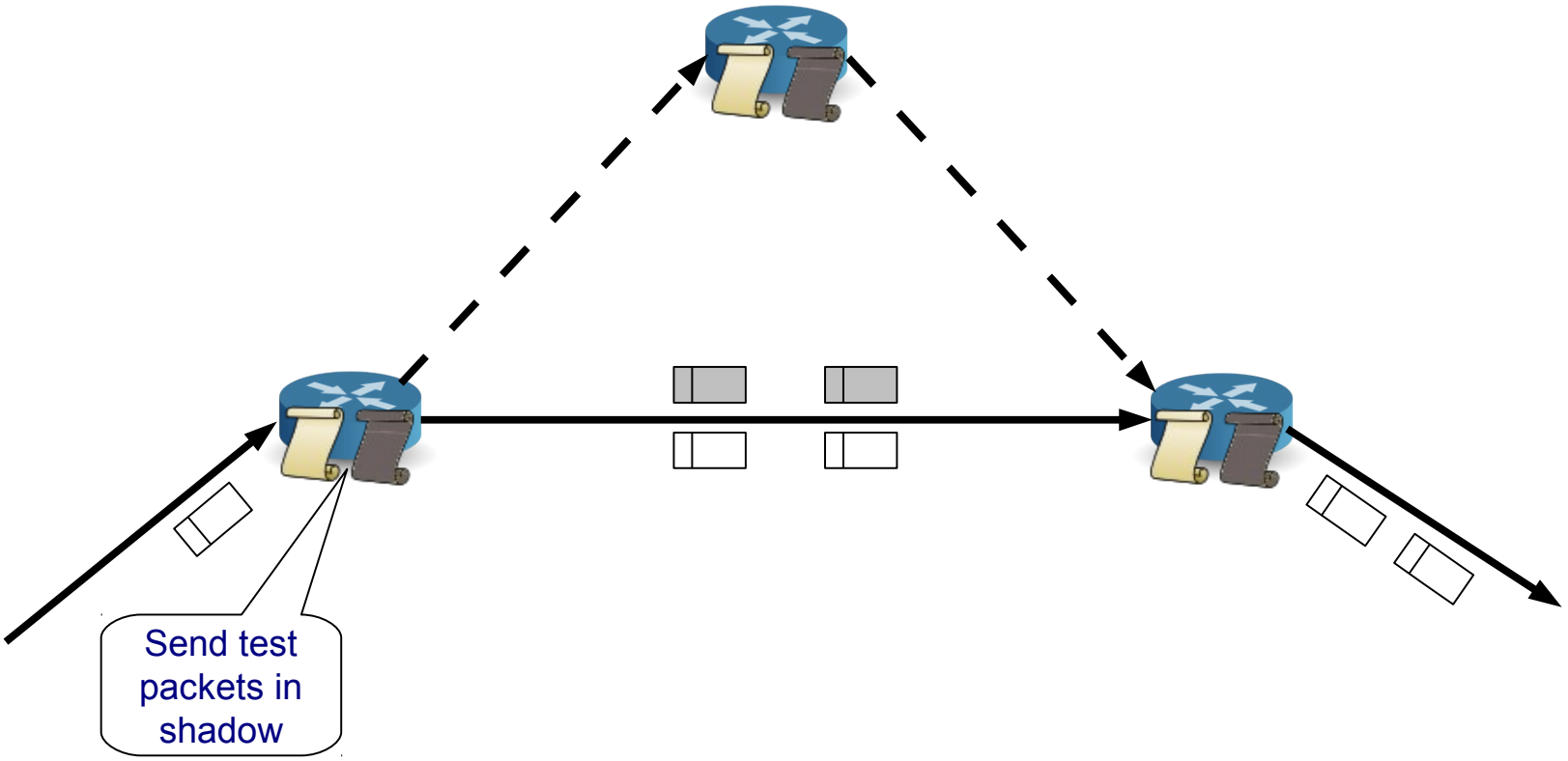
- Routing parameters
- ACLs
- Interface parameters
- VPNs
- QoS parameters
- ...



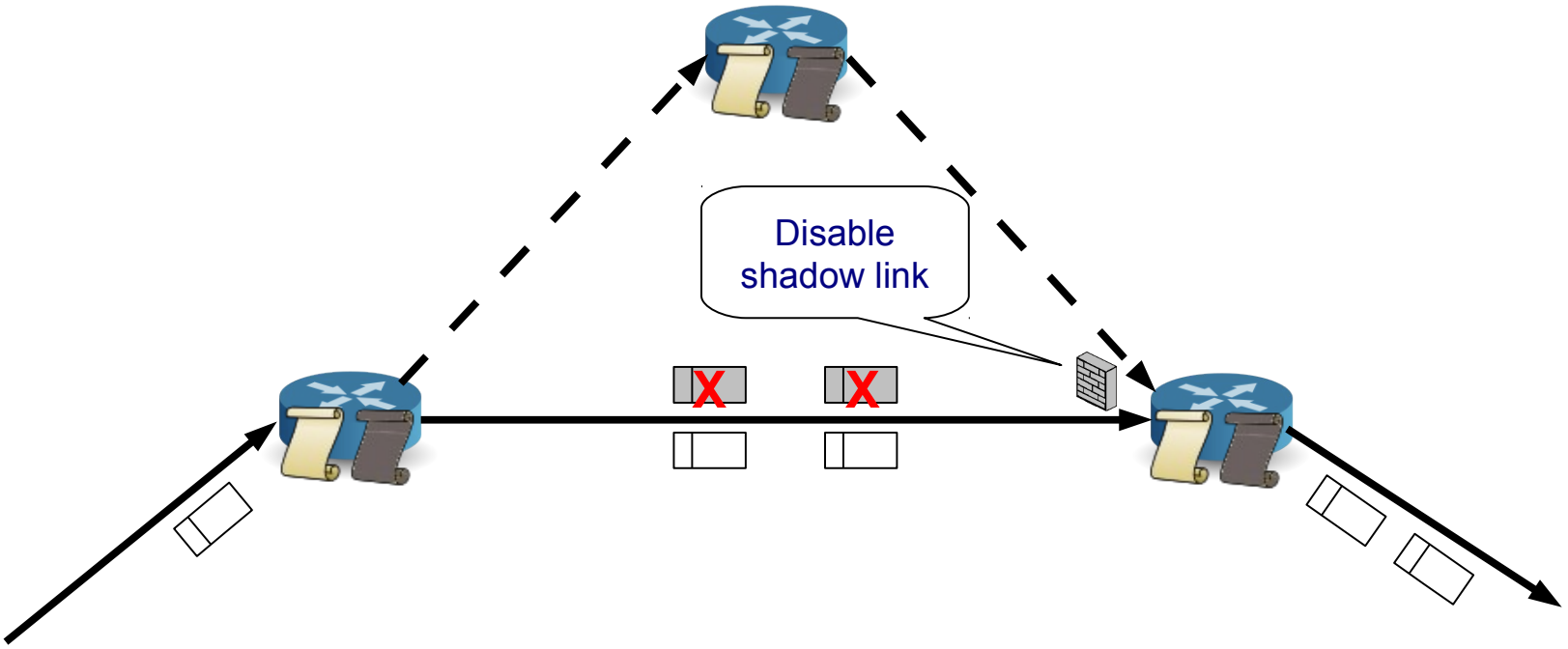
Example Usage Scenario: Backup Path Verification



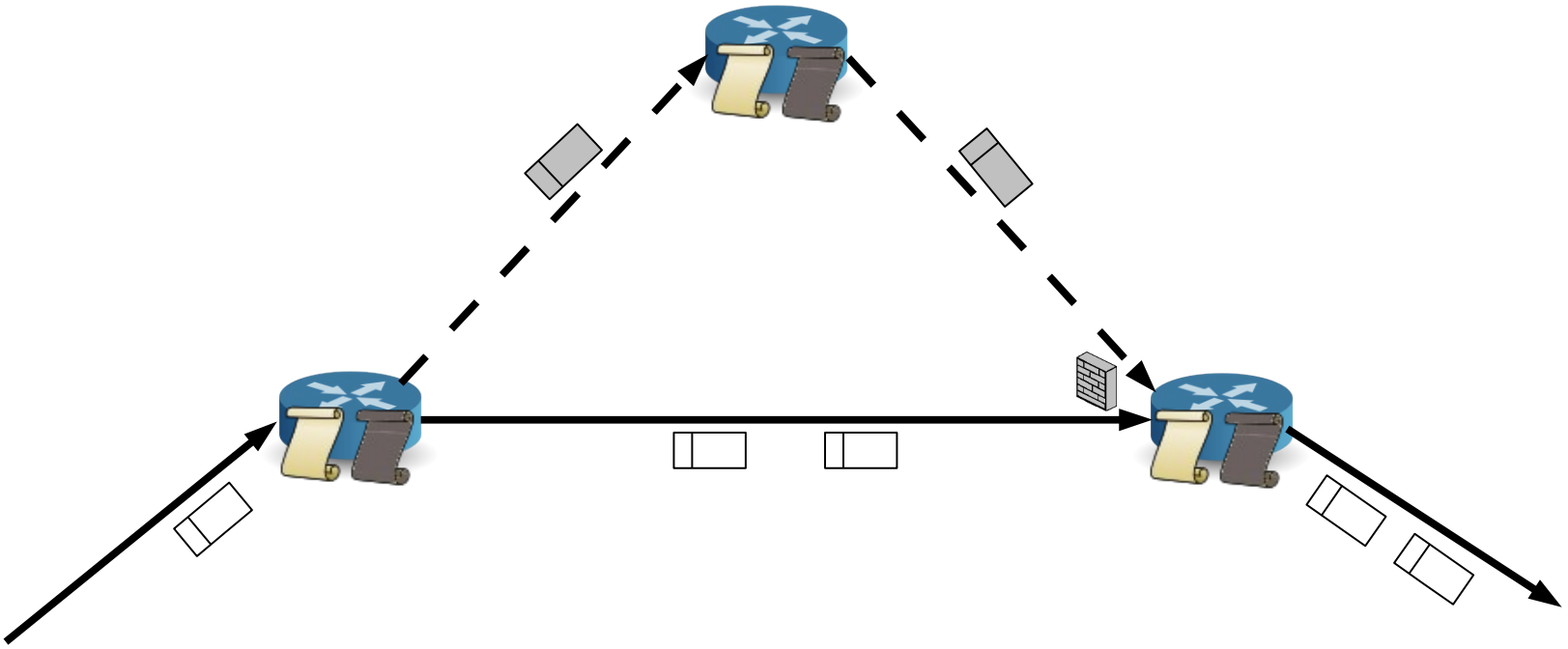
Example Usage Scenario: Backup Path Verification



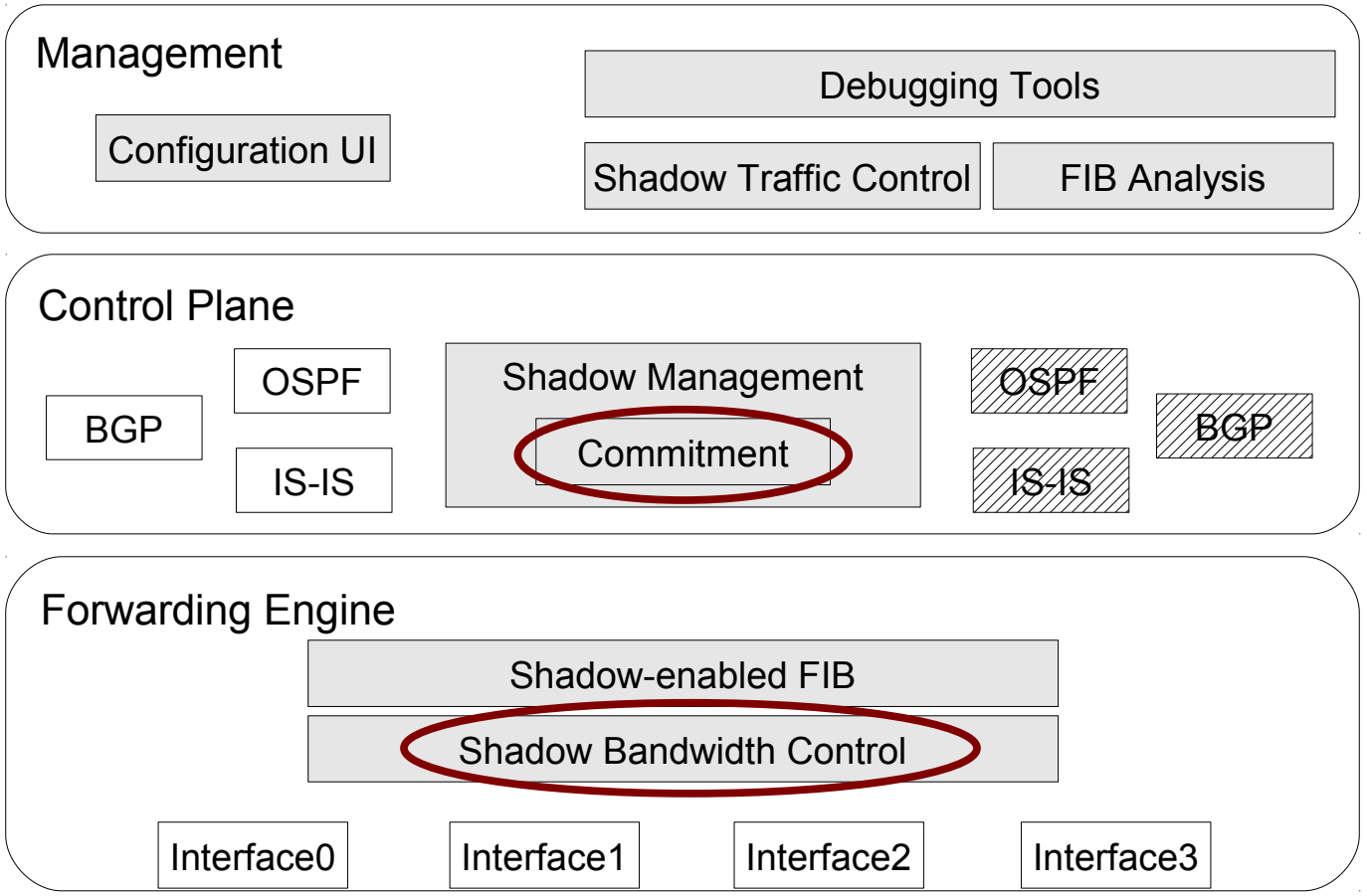
Example Usage Scenario: Backup Path Verification



Example Usage Scenario: Backup Path Verification



Design and Architecture



Shadow Bandwidth Control

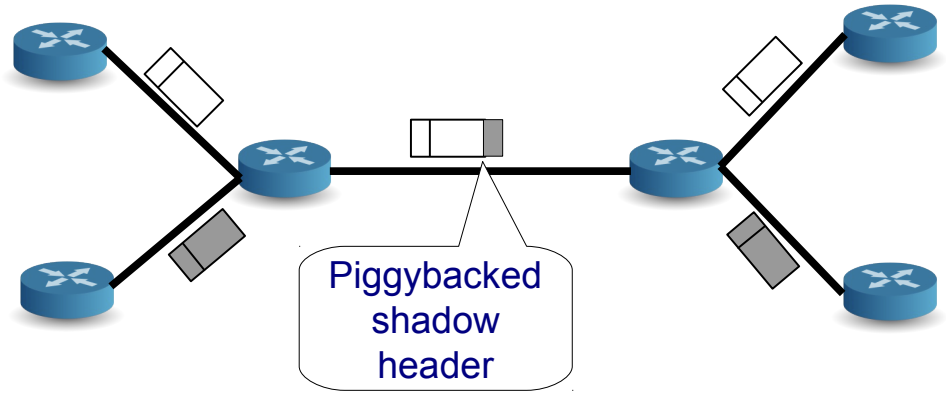
Requirements

- Minimal impact on production traffic
- Accurate performance measurements of shadow configuration

Supported Modes

- Priority, Bandwidth Partitioning
- *Packet Cancellation*

In many network performance testing scenarios, only payload size matters



Commitment

Objectives

- Smoothly swap production and shadow across network
  - Eliminate effects of reconvergence due to config changes
- Easy to swap back

Issue

- Shadow bit within packet determines which FIB to use
- Routers swap FIBs asynchronously
- Inconsistent FIBs applied on the path

- → We use tags to achieve consistency

Implementation

Kernel-level (based on Linux 2.6.22.9)

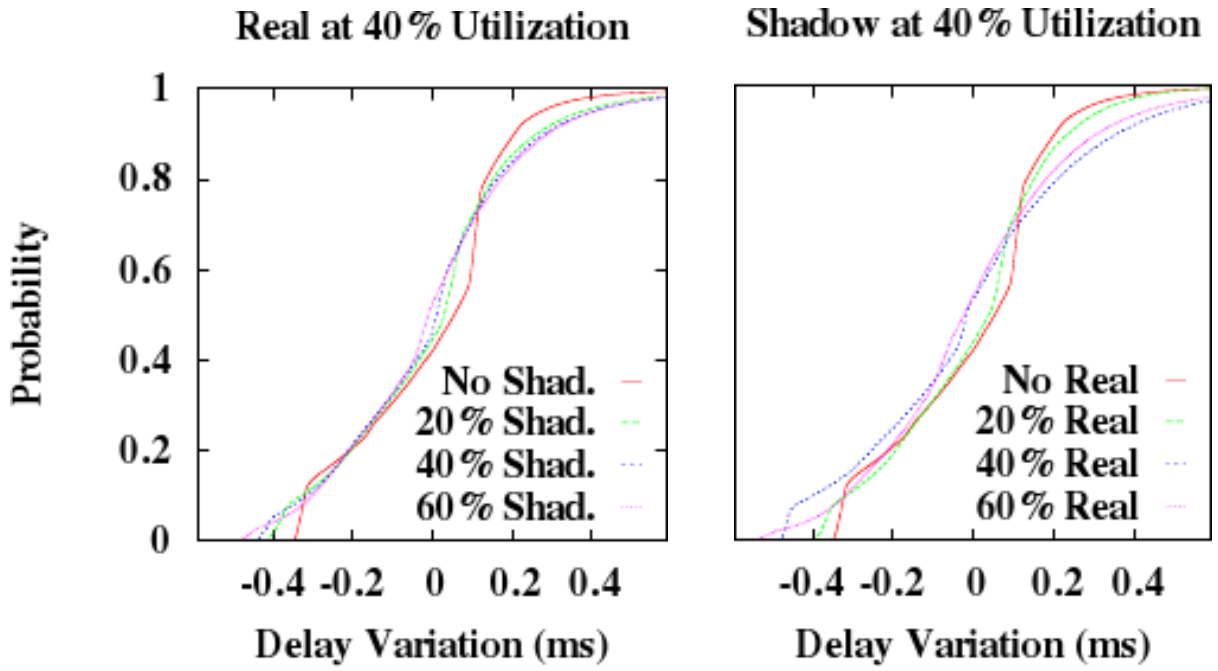
- TCP/IP stack support
- FIB management
- Commitment hooks
- Packet cancellation

Tools

- Transparent software router support (Quagga + XORP)
- Full commitment protocol
- Configuration UI (command-line based)

Evaluated on Emulab (3Ghz HT CPUs)

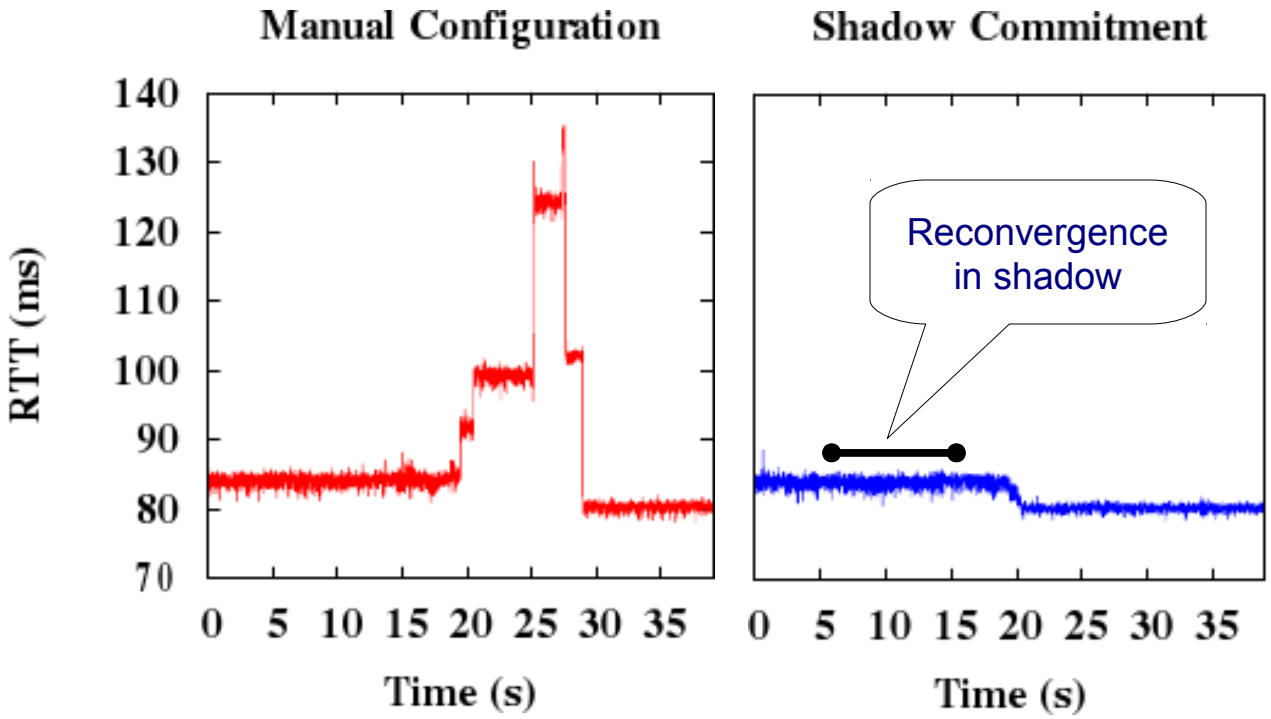
Evaluation: Packet Cancellation



Limited interaction of production and shadow

- Intersecting production and shadow flows
  - CAIDA traces
- Vary flow utilizations

Evaluation: Commitment



Applying OSPF link-weight changes

- Abilene topology with 3 external peers
  - Configs translated to Quagga syntax
  - Abilene BGP dumps

Conclusion and Future Directions

Contributions

- A Dual-System Architecture supporting testing as basic capability on a production infrastructure
- Architecture is applied in two diverse contexts
  - P2P live streaming and network configuration management

Future Directions

- Incremental deployment
  - What if part of my production infrastructure is outside of the boundary?
- Integration with online debugging and verification techniques
  - Can we stop and inspect test system?
- Application in other contexts
  - Examples: Video-on-demand, CDN infrastructures

Research Output

Publications

- R. Alimi, C. Tian, Y.R. Yang, D. Zhang, “PEAC: Performance Experimentation as a Capability in Production Internet Live Streaming”, *Under submission*
- L.E. Li, R. Alimi, D. Shen, H. Viswanathan, Y.R. Yang, “A General Algorithm for Interference Alignment and Cancellation in Wireless Networks”, in Infocom 2010
- Y. Wang, H. Wang, A. Mahimkar, R. Alimi, Y. Zhang, L. Qiu, Y.R. Yang, “R3: resilient routing reconfiguration”, In Sigcomm 2010
- L.E. Li, R. Alimi, R. Ramjee, H. Viswanathan, Y.R. Yang, “muNet: Harnessing Multiuser Capacity in Wireless Mesh Networks”, In Infocom 2009
- R. Alimi, L.E. Li, R. Ramjee, H. Viswanathan, Y.R. Yang, “iPack: in-Network Packet Mixing for High Throughput Wireless Mesh Networks”, In Infocom 2008
- R. Alimi, Y. Wang, Y.R. Yang, “Shadow configuration as a network management primitive”, In Sigcomm 2008
- L.E. Li, R. Alimi, R. Ramjee, J. Shi, Y. Sun, H. Viswanathan, Y.R. Yang, “Superposition coding for wireless mesh networks”, Extended abstract, In Mobicom 2007

Other Projects

- P4P: Provider Portal for Applications
- DECADE: Open Content Distribution using Data Lockers

Thank you!

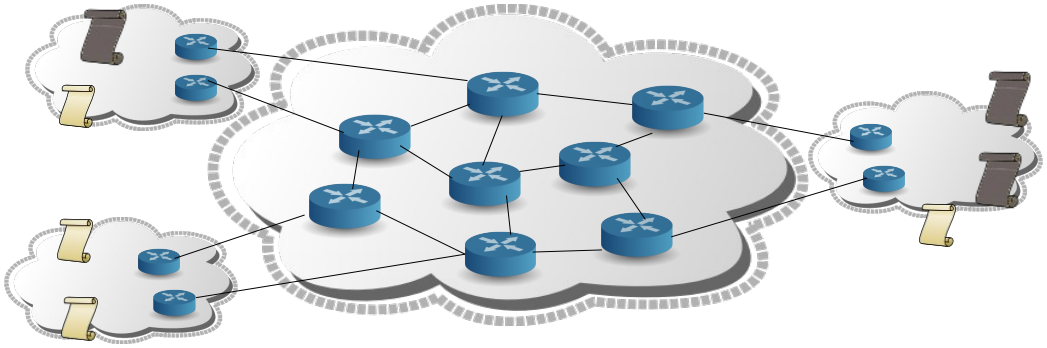
Questions?

Backup Slides

User Partitioning

Designate “test” users

- Use only selected users
- Measure effects directly



Benefits

- Real system running
- *Real environment*

Limitations

- Possible disruptions to users
- Difficult to control testing scenarios

PEAC

Use Cases

Regression Tests with User Performance

- Define tests based on expected performance
- Run tests before new release

Parameter Tuning

- Parallel tests with different parameters
- Factor analysis

Algorithm/Feature Testing

- Test in real network environments
- Complementary to modeling, simulation, analysis

Scale-invariant Streaming

For a class of algorithms and network settings, if we

- scale channel (streaming) rate by α (e.g., 1/5)
- scale the upload capacities of end-hosts by same α

then certain performance metrics remain unchanged

- Don't need to know relationship between performance and input parameters
- Easier to protect against disruption with small α

Certain (common) settings are not scale-invariant

- Example: rate control with slow-start, bottlenecks within network

Implementing ATR

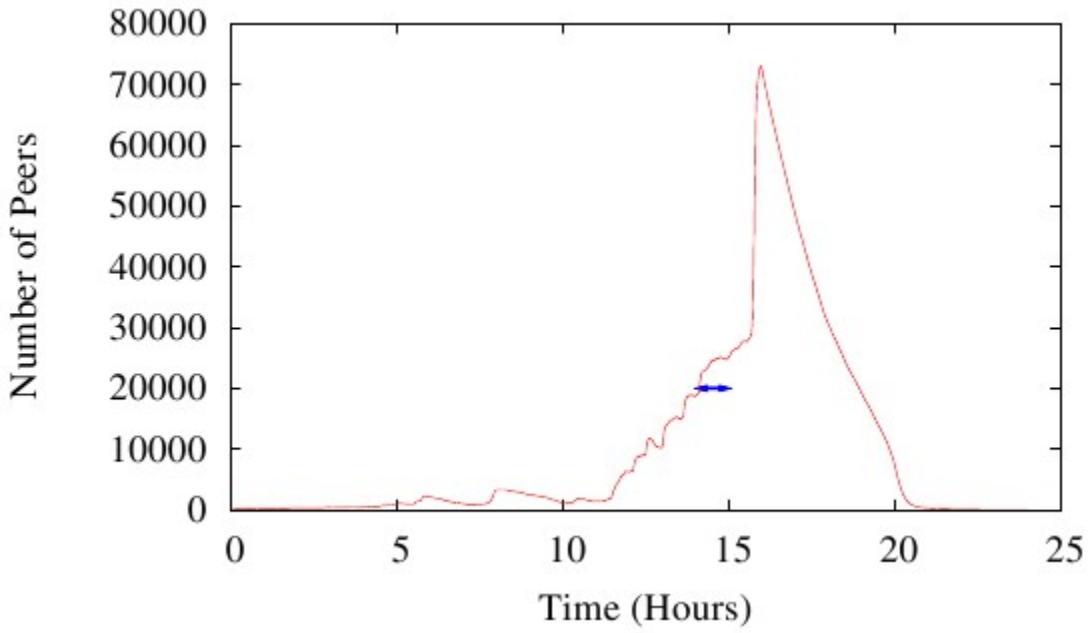
Reallocate tasks from test to production

- But.. we wish to treat systems as black-box

How does ATR Scheduler allocate tasks?

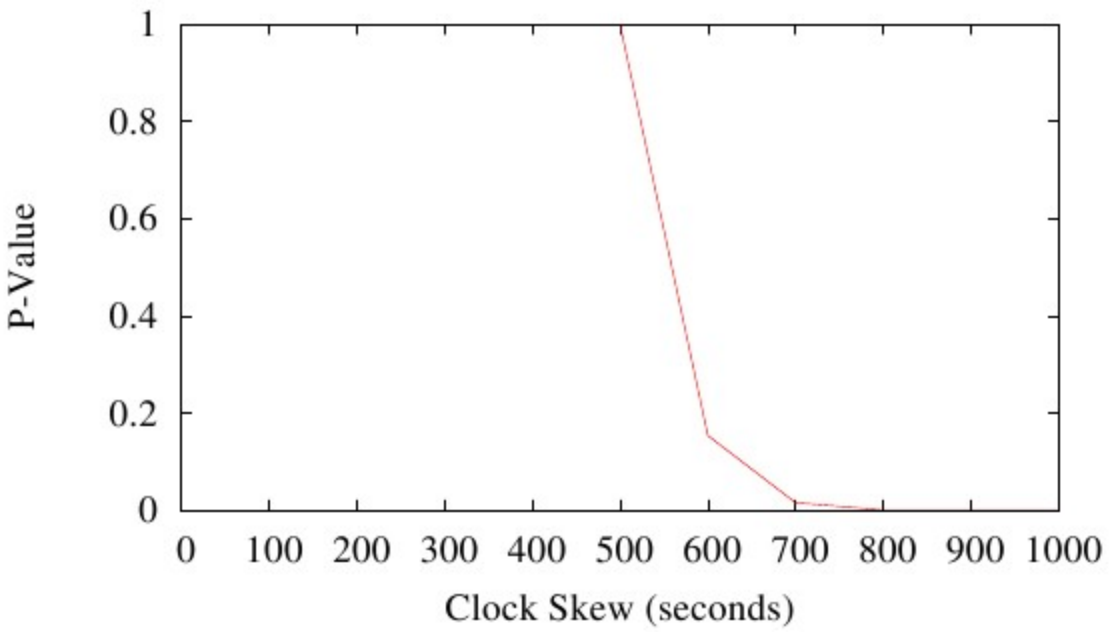
Buffer window itself is used as control API

- *getPlaypointRange()*
- *setBufferWindowPos(pos)*

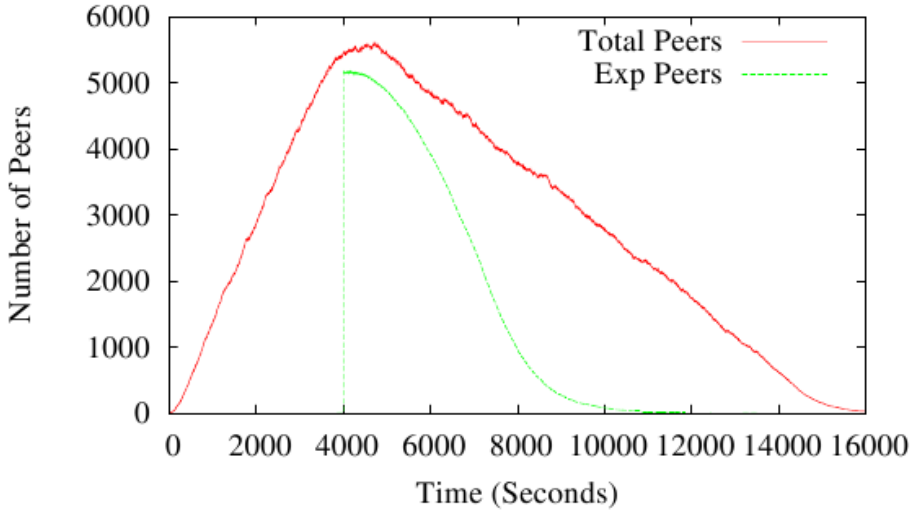


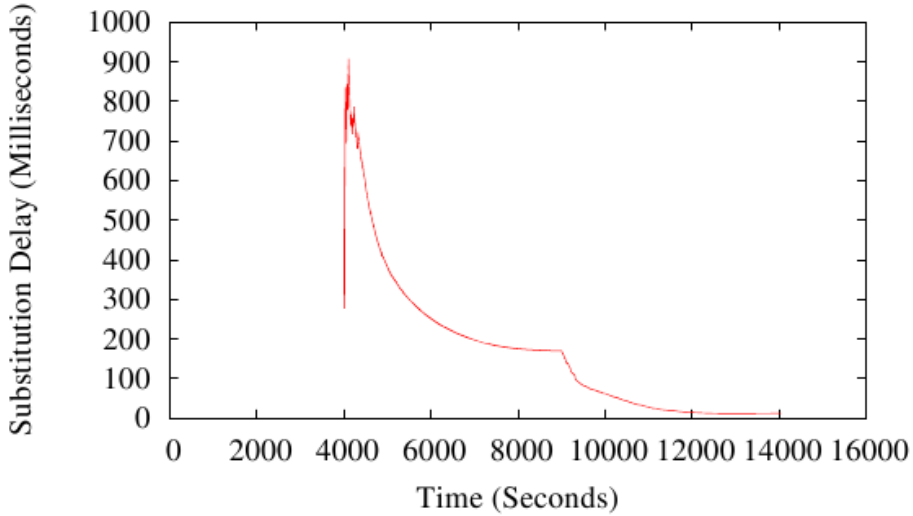
Opportunities to trigger 4-hour, 20,000-peer experiment in PPLive's HN Satellite channel

Accuracy of Generated Arrival Behavior



Chi-square goodness-of-fit test according to clock-skew for generated arrival behavior for baseball game with about 60,000 concurrent peers





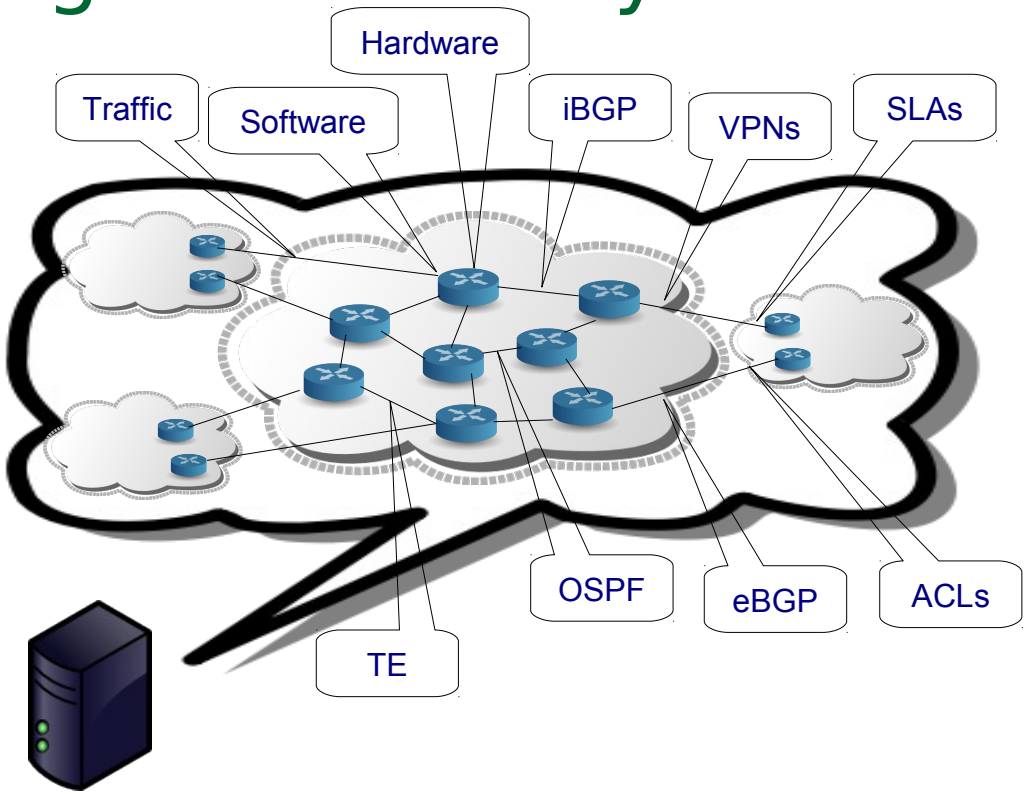
Substitution delay with user-initiated departures

ShadowNet

Configuration Management Today

Simulation & Analysis

- Depend on simplified models
  - Network structure
  - Hardware and software
- Limited scalability
- Hard to access real traffic



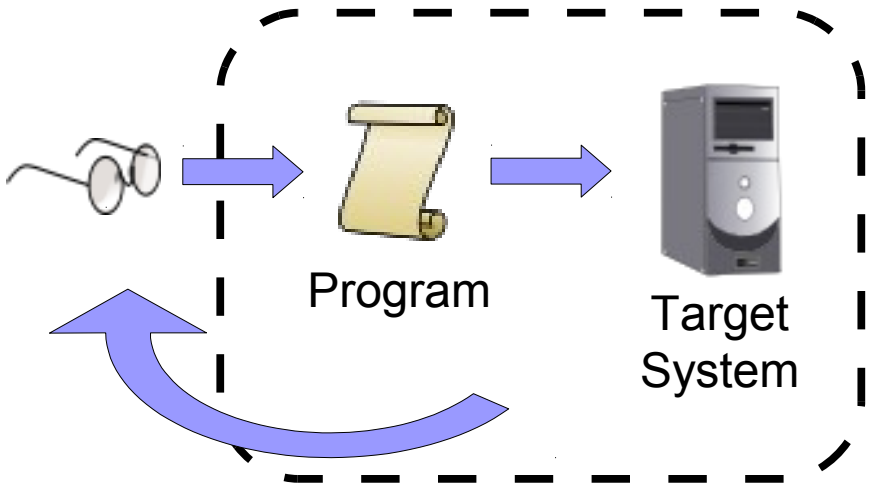
Test networks

- Can be prohibitively expensive

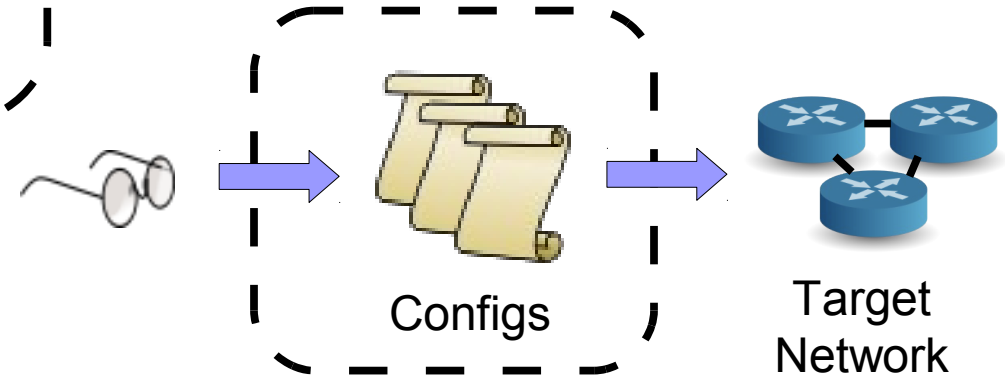
Why are these not enough?

Analogy with Programming

Programming

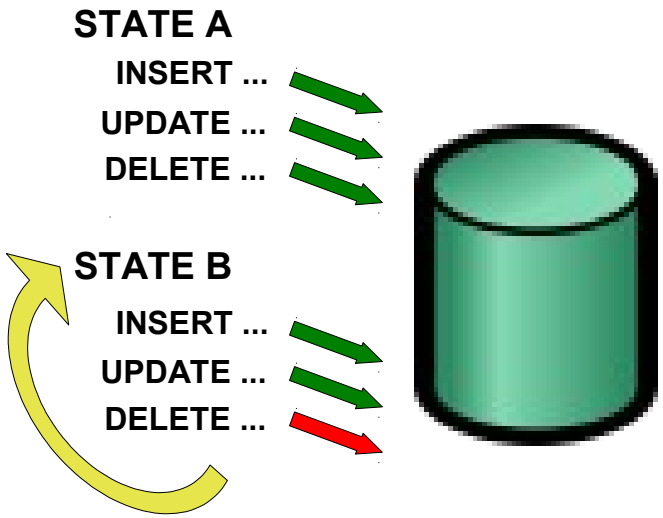


Network Management

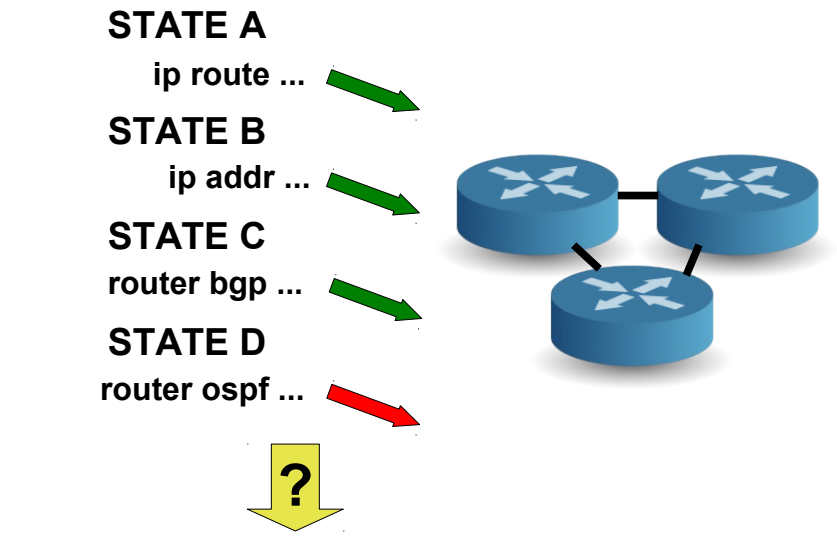


Analogy with Databases

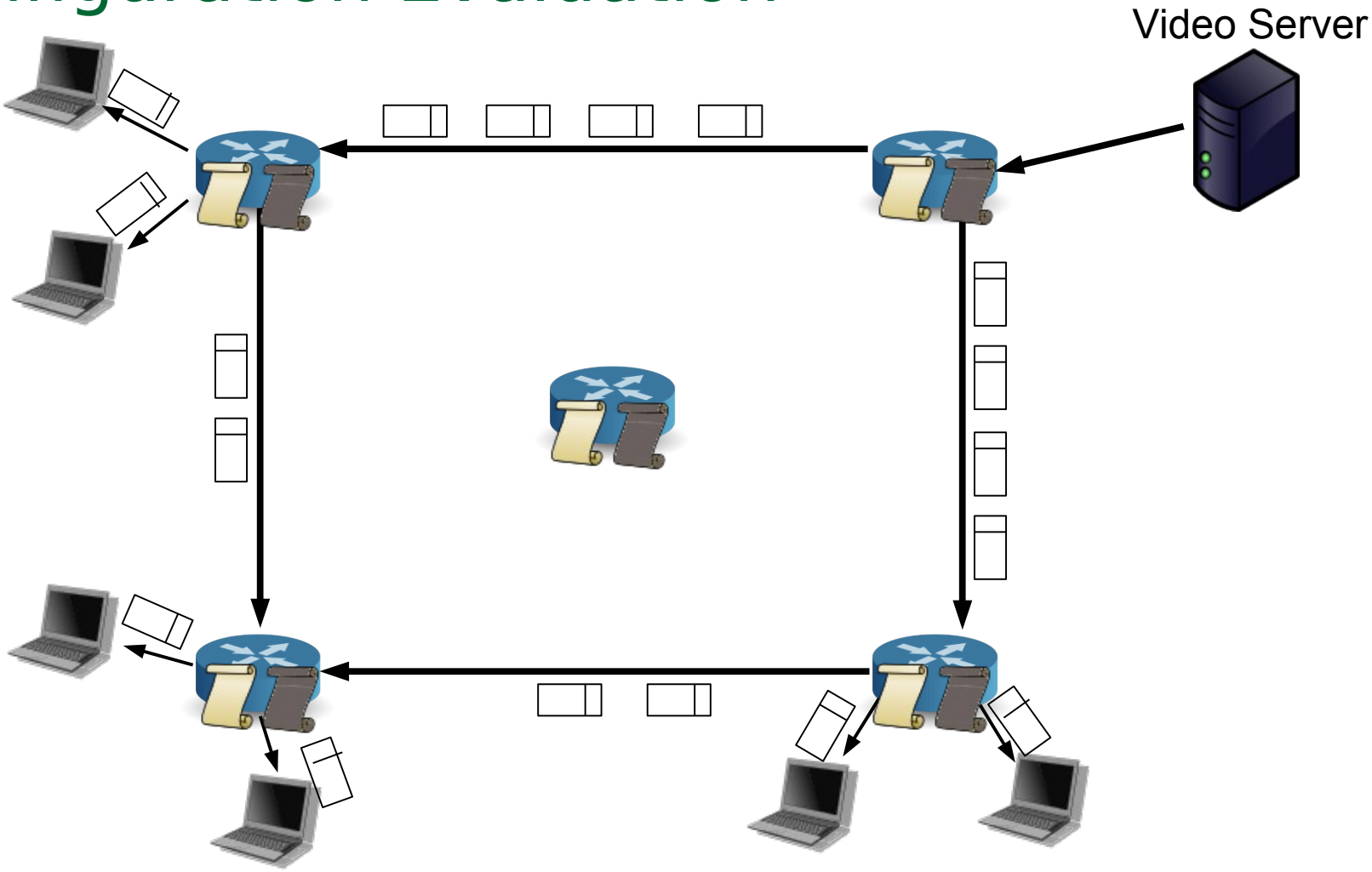
Databases



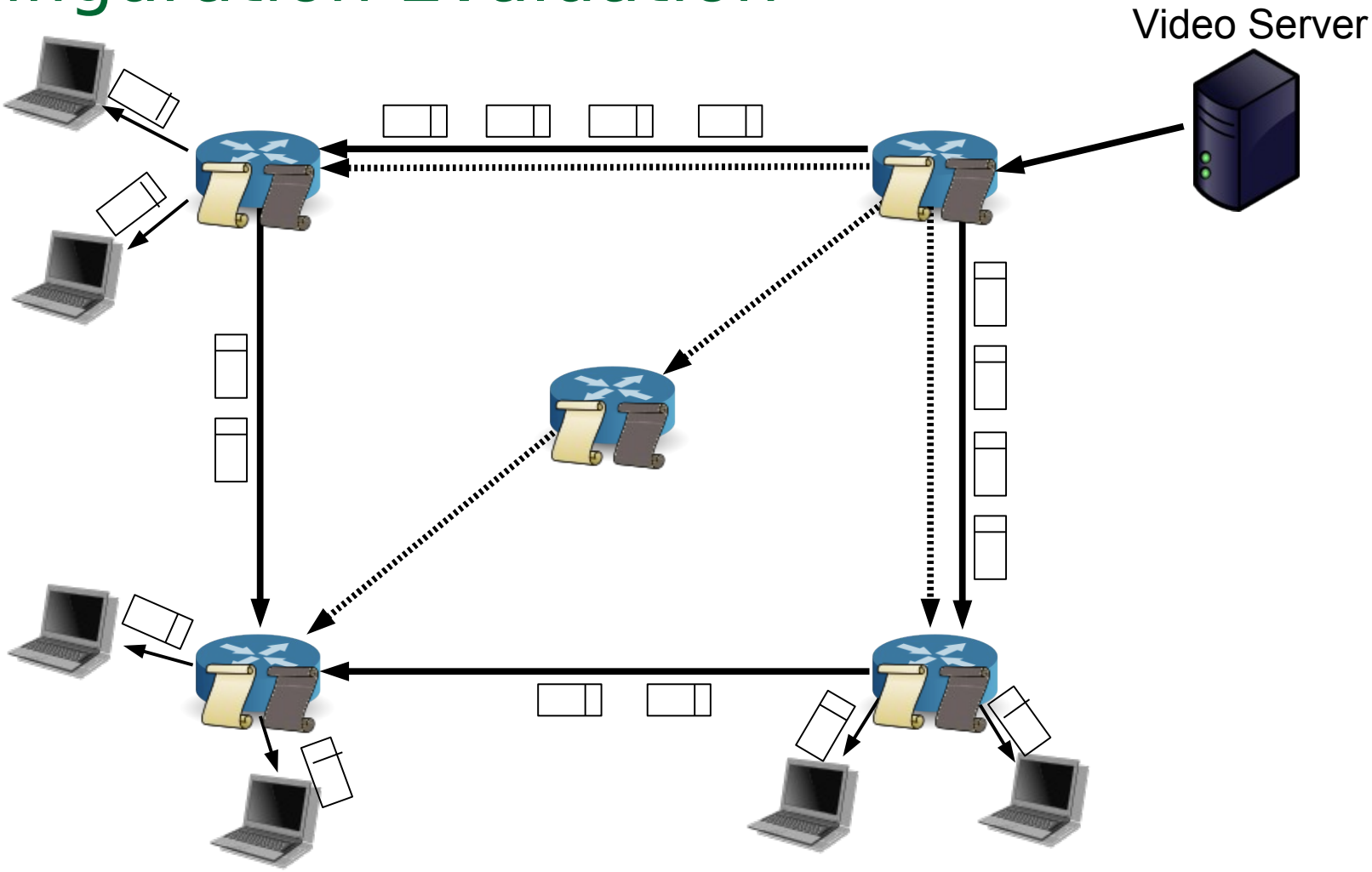
Network Management



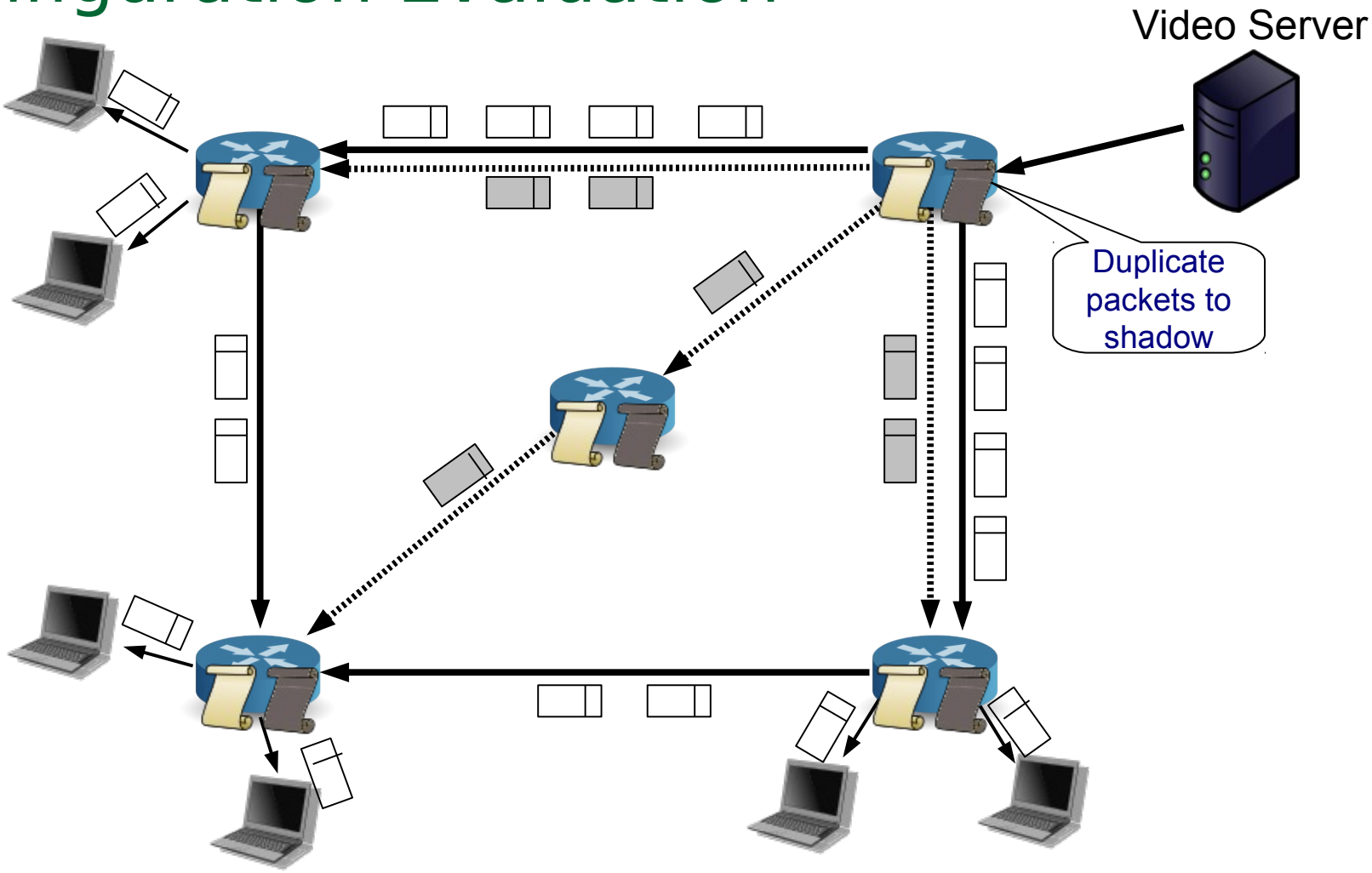
Example Usage Scenario: Configuration Evaluation



Example Usage Scenario: Configuration Evaluation



Example Usage Scenario: Configuration Evaluation

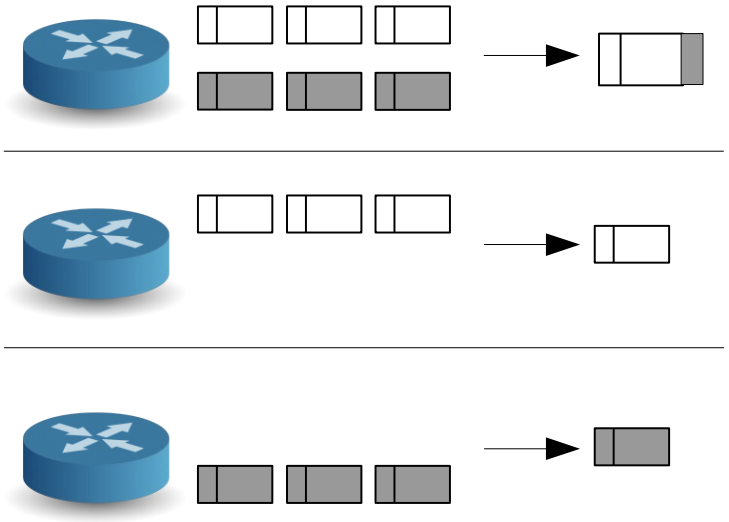


Packet Cancellation Details

Output interface maintains real and shadow queues

- Q_r and Q_s

```
pktsched() – packet cancellation and scheduling.
01. if not empty(Q_r) then
02.     p ← dequeue(Q_r) // Select real packet
03.     // Append shadow packet headers
04.     for 1...MAX_CANCELLABLE do
05.         if not virtual_clock_expired(peek(Q_s))
06.             break
07.         p ← append(p, ip_hdr(dequeue(Q_s)))
08.     endfor
09.     transmit(p)
10. elseif not empty(Q_s) then
11.     // Send shadow packet if available
12.     if virtual_clock_expired(peek(Q_s))
13.         transmit(dequeue(Q_s))
14. endif
```

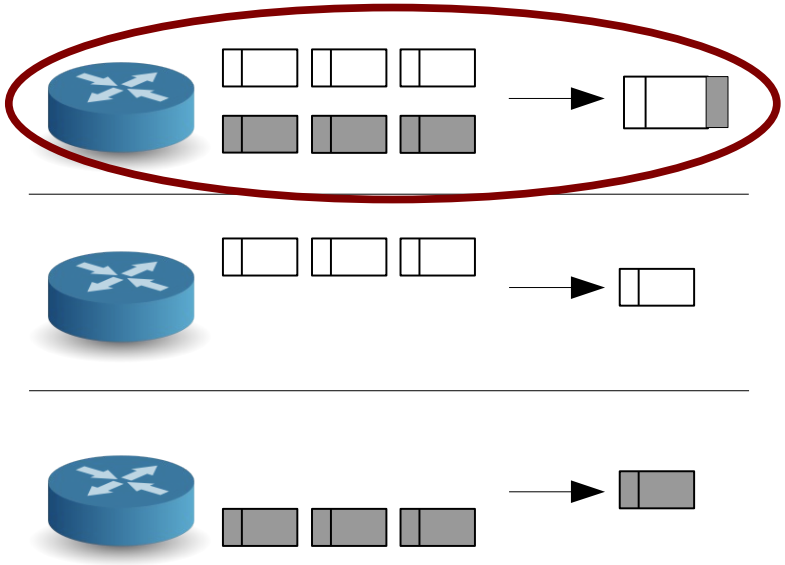


Packet Cancellation Details

Output interface maintains real and shadow queues

- Q_r and Q_s

```
pktsched() – packet cancellation and scheduling.
01. if not empty(Q_r) then
02.     p ← dequeue(Q_r) // Select real packet
03.     // Append shadow packet headers
04.     for 1...MAX_CANCELLABLE do
05.         if not virtual_clock_expired(peek(Q_s))
06.             break
07.         p ← append(p, ip_hdr(dequeue(Q_s)))
08.     endfor
09.     transmit(p)
10. elseif not empty(Q_s) then
11.     // Send shadow packet if available
12.     if virtual_clock_expired(peek(Q_s))
13.         transmit(dequeue(Q_s))
14. endif
```

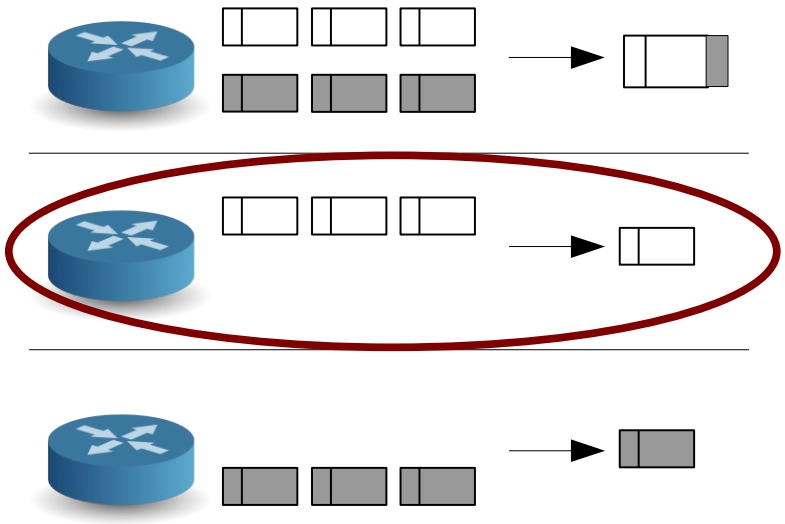


Packet Cancellation Details

Output interface maintains real and shadow queues

- Q_r and Q_s

```
pktsched() – packet cancellation and scheduling.
01. if not empty(Qr) then
02.     p ← dequeue(Qr) // Select real packet
03.     // Append shadow packet headers
04.     for 1...MAX_CANCELLABLE do
05.         if not virtual_clock_expired(peek(Qs))
06.             break
07.         p ← append(p, ip_hdr(dequeue(Qs)))
08.     endfor
09.     transmit(p)
10. elseif not empty(Qs) then
11.     // Send shadow packet if available
12.     if virtual_clock_expired(peek(Qs))
13.         transmit(dequeue(Qs))
14. endif
```

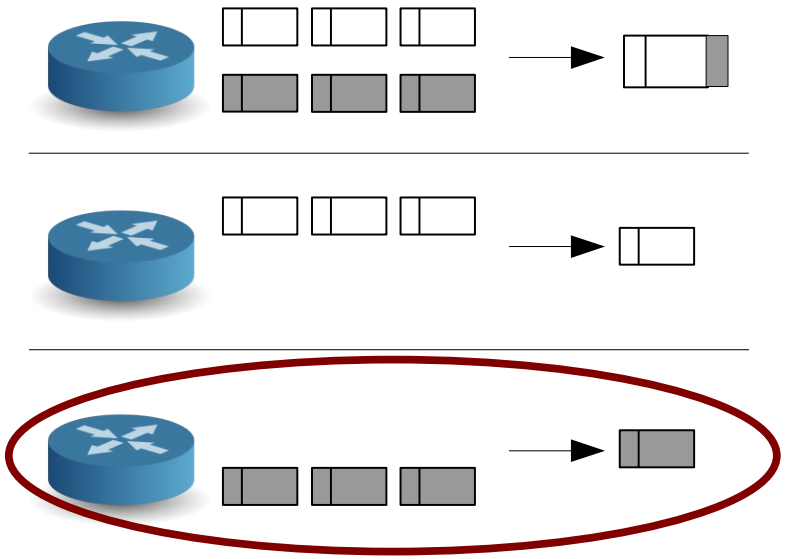


Packet Cancellation Details

Output interface maintains real and shadow queues

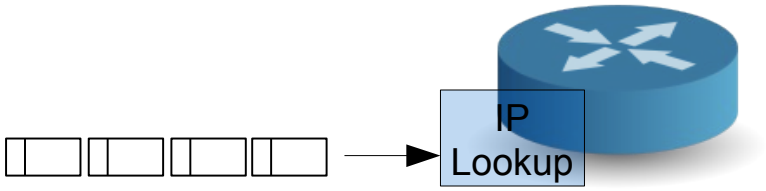
- Q_r and Q_s

```
pktsched() – packet cancellation and scheduling.
01. if not empty(Q_r) then
02.     p ← dequeue(Q_r) // Select real packet
03.     // Append shadow packet headers
04.     for 1...MAX_CANCELLABLE do
05.         if not virtual_clock_expired(peek(Q_s))
06.             break
07.         p ← append(p, ip_hdr(dequeue(Q_s)))
08.     endfor
09.     transmit(p)
10. elseif not empty(Q_s) then
11.     // Send shadow packet if available
12.     if virtual_clock_expired(peek(Q_s))
13.         transmit(dequeue(Q_s))
14. endif
```

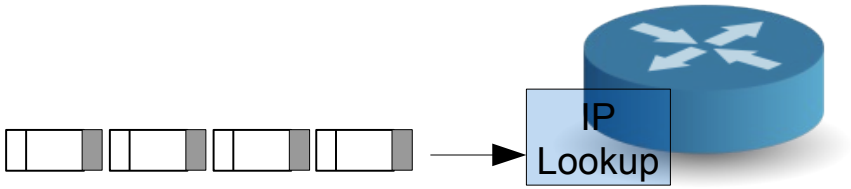


Forwarding Overhead

Without Packet Cancellation:



With Packet Cancellation:



Cancellation may require routers to process more packets.
Can routers support it?

Forwarding Overhead Analysis

Routers can be designed for worst-case

- L : Link speed
- K_{min} : Minimum packet size
- Router supports $\alpha \frac{L}{K_{min}}$ packets per second

Load typically measured by link utilization

- α_r : Utilization due to real traffic (packet sizes k_r)
- α_s : Utilization due to shadow traffic (packet sizes k_s)

We require:

$$\mathbb{E}\left[\frac{\alpha_r L}{k_r}\right] + \mathbb{E}\left[\frac{\alpha_s L}{k_s}\right] < \alpha \frac{L}{K_{min}}$$

Forwarding Overhead Analysis

Routers can be designed for worst-case

- L : Link speed
- K_{min} : Minimum packet size
- Router supports $\alpha \frac{L}{K_{min}}$ packets per second

Load typically measured by link utilization

- α_r : Utilization due to real traffic (packet sizes k_r)
- α_s : Utilization due to shadow traffic (packet sizes k_s)

We require:

$$\mathbb{E}\left[\frac{\alpha_r L}{k_r}\right] + \mathbb{E}\left[\frac{\alpha_s L}{k_s}\right] < \alpha \frac{L}{K_{min}}$$

Example:
With α = 70%, and 80% real traffic utilization
*Support up to **75% shadow traffic utilization***

Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

Basic algorithm has 4 phases

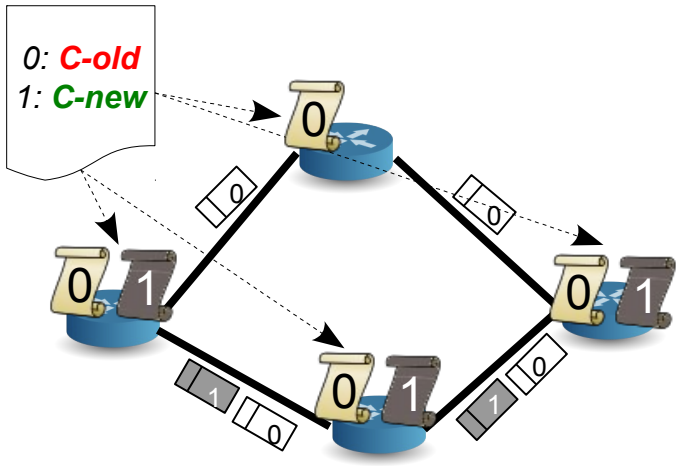
Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

Basic algorithm has 4 phases

- Distribute tags for each config
  - **C-old** for current real config
  - **C-new** for current shadow config



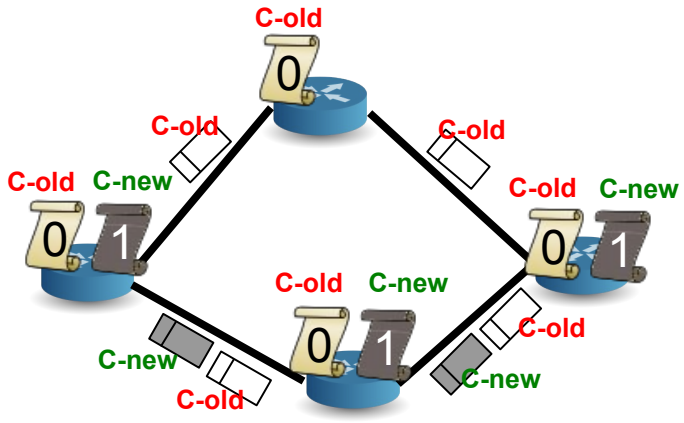
Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

Basic algorithm has 4 phases

- Distribute tags for each config
  - **C-old** for current real config
  - **C-new** for current shadow config
- Routers mark packets with tags
  - Packets forwarded according to tags



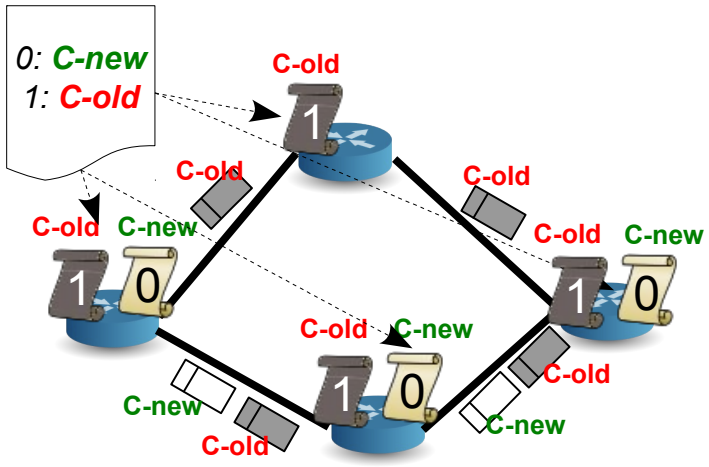
Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

Basic algorithm has 4 phases

- Distribute tags for each config
  - **C-old** for current real config
  - **C-new** for current shadow config
- Routers mark packets with tags
  - Packets forwarded according to tags
- Swap configs (tags still valid)



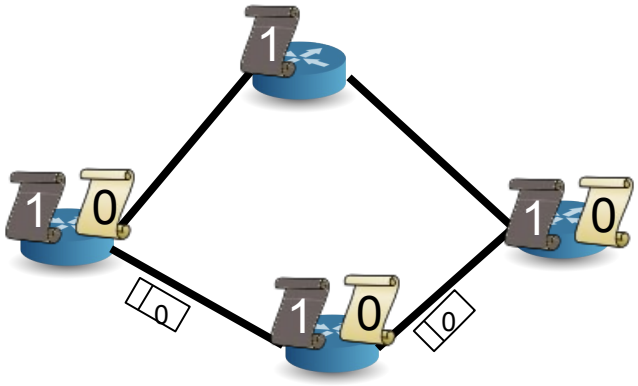
Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

Basic algorithm has 4 phases

- Distribute tags for each config
  - **C-old** for current real config
  - **C-new** for current shadow config
- Routers mark packets with tags
  - Packets forwarded according to tags
- Swap configs (tags still valid)
- Remove tags from packets
  - Resume use of shadow bit



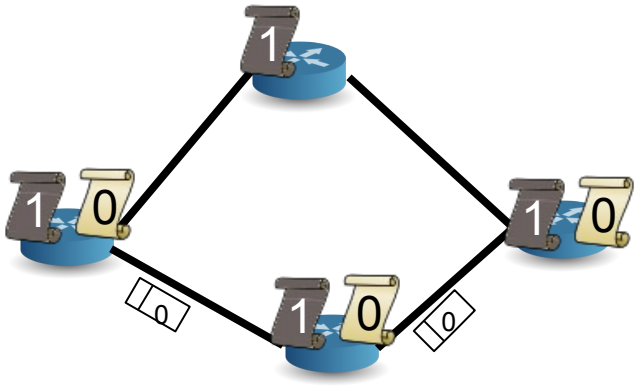
Commitment Protocol

Idea: Use tags to achieve consistency

- Temporary identifiers

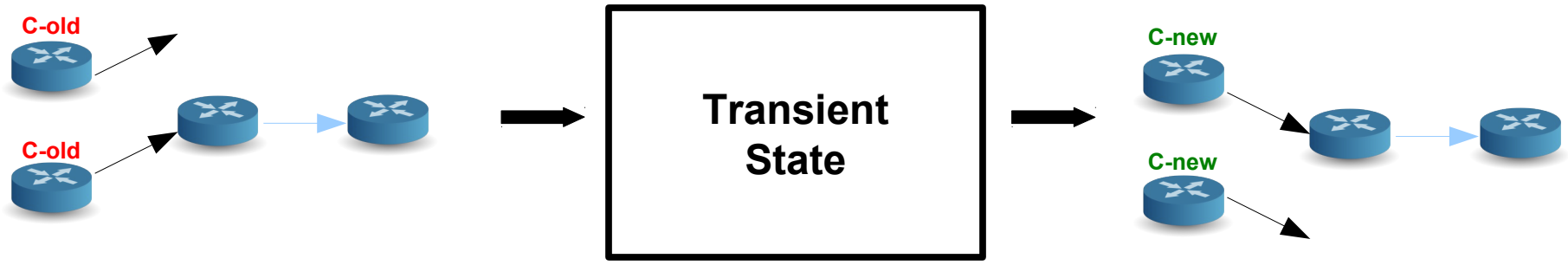
Basic algorithm has 4 phases

- Distribute tags for each config
  - **C-old** for current real config
  - **C-new** for current shadow config
- Routers mark packets with tags
  - Packets forwarded according to tags
- Swap configs (tags still valid)
- Remove tags from packets
  - Resume use of shadow bit



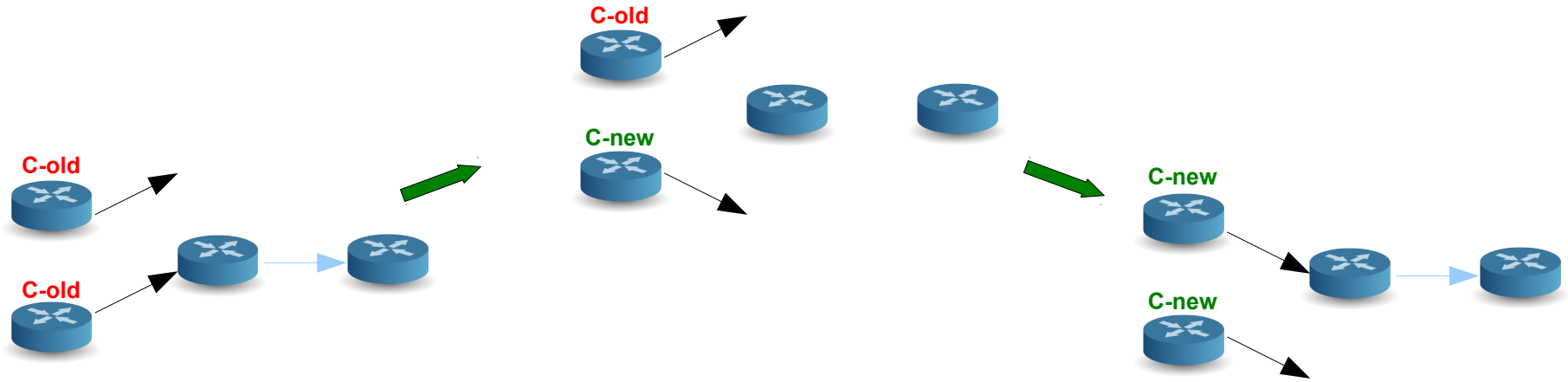
Transient States

Definition: State in which some packets use **C-old** and others use **C-new**.



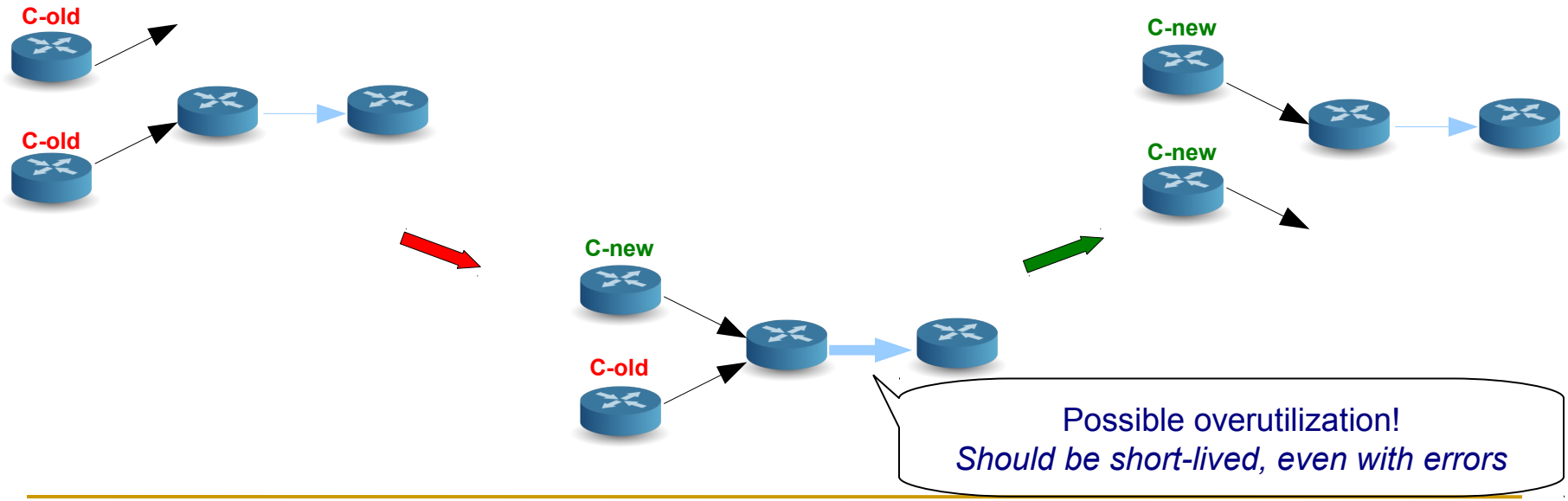
Transient States

Definition: State in which some packets use **C-old** and others use **C-new**.



Transient States

Definition: State in which some packets use **C-old** and others use **C-new**.

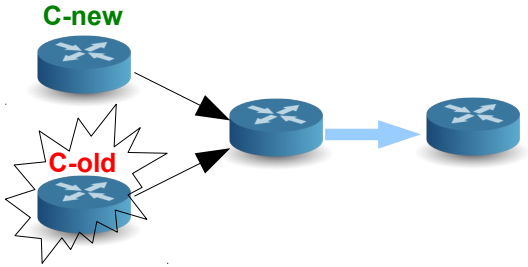


Error Recovery During Swap

If ACK missing from at least one router, two cases:

(a) Router completed SWAP but ACK not sent

(b) Router did not complete SWAP ***Transient State***



Error Recovery During Swap

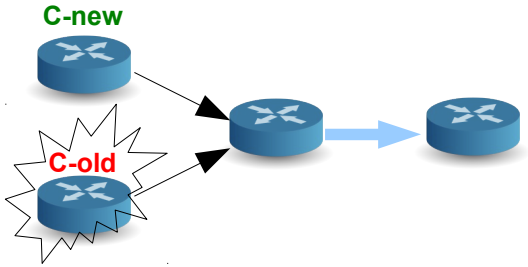
If ACK missing from at least one router, two cases:

(a) Router completed SWAP but ACK not sent

(b) Router did not complete SWAP ***Transient State***

Detect (b) and rollback quickly

- Querying router directly may be impossible



Error Recovery During Swap

If ACK missing from at least one router, two cases:

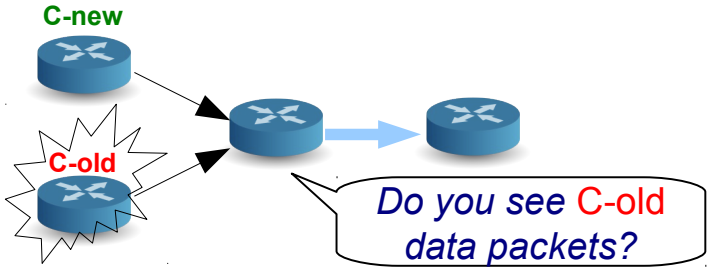
(a) Router completed SWAP but ACK not sent

(b) Router did not complete SWAP ***Transient State***

Detect (b) and rollback quickly

- Querying router directly may be impossible

Solution: Ask neighboring routers

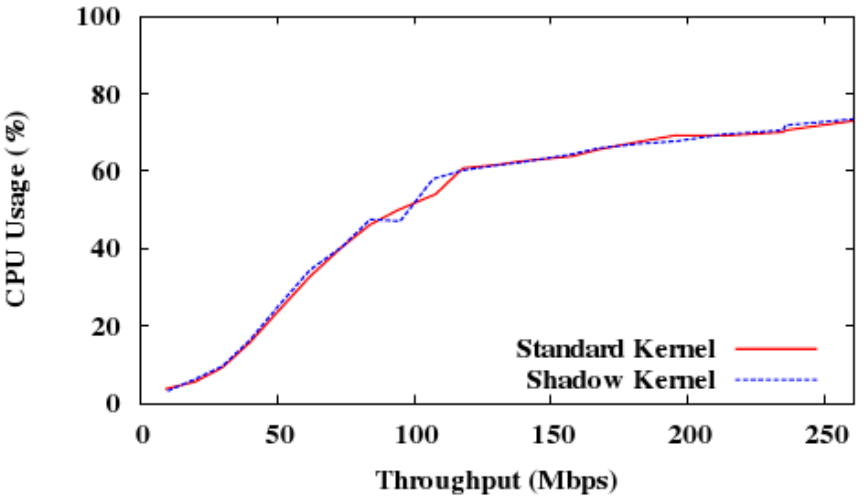


If YES:
Case (b): rollback other routers
Otherwise,
Case (a): no transient state

Evaluation: CPU Overhead

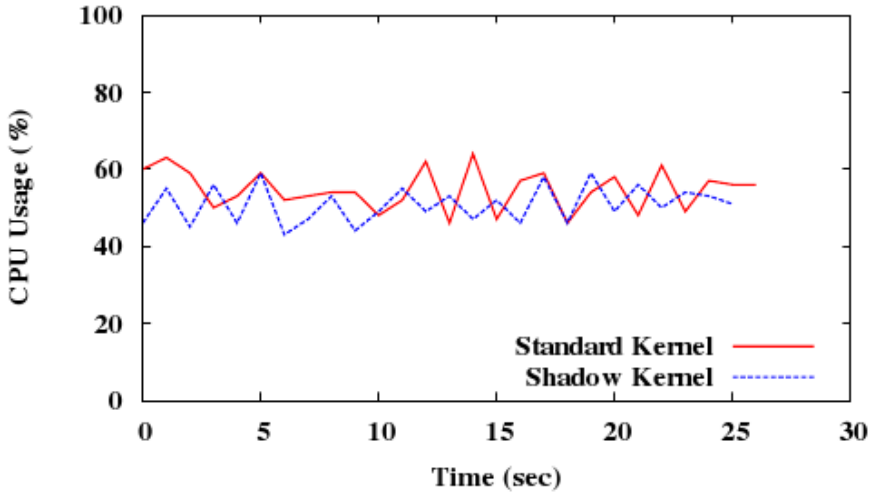
Static FIB

- 300B pkts
- No route caching



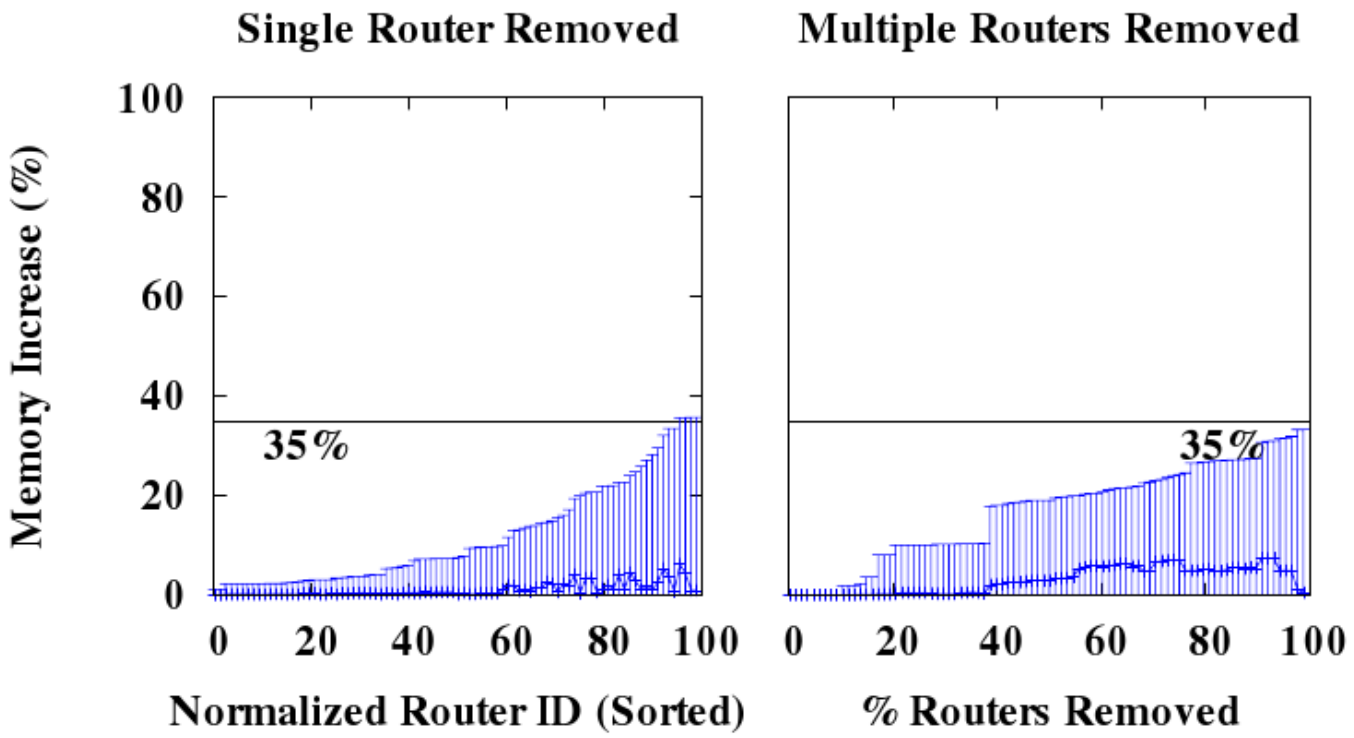
With FIB updates

- 300B pkts @ 100Mbps
- 1-100 updates/sec
- No route caching

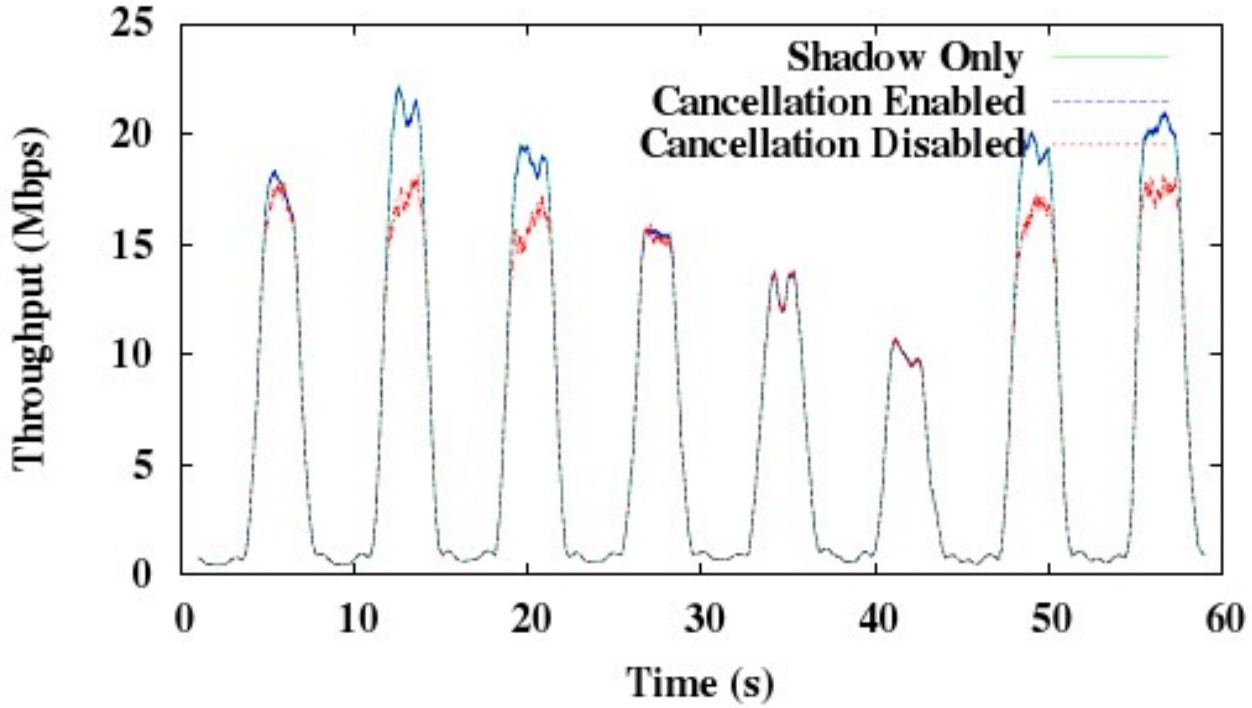


Evaluation: Memory Overhead

FIB storage overhead for US Tier-1 ISP



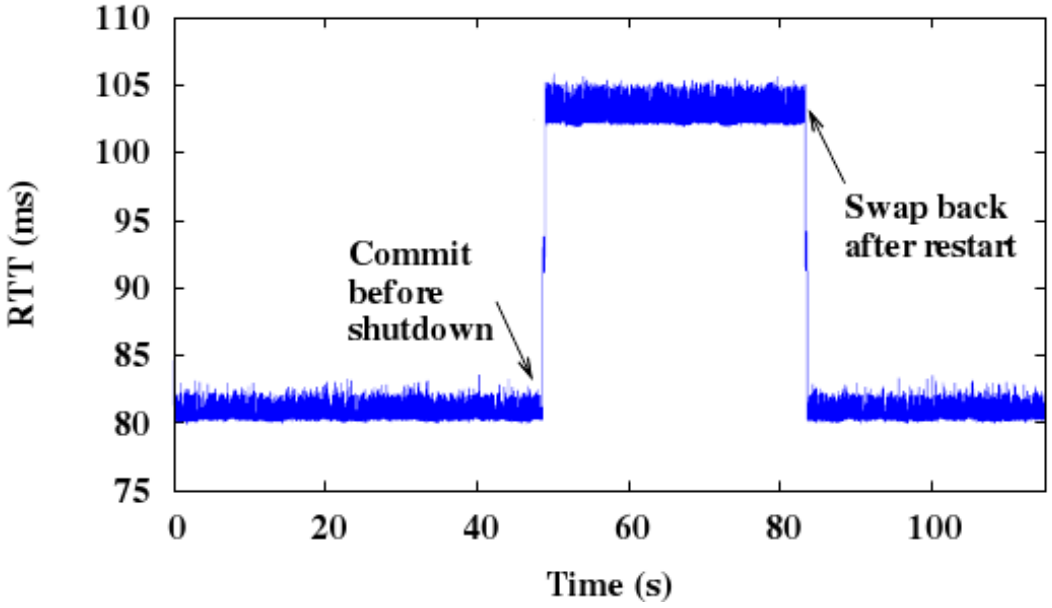
Evaluation: Packet Cancellation



Accurate streaming throughput measurement

- Abilene topology
- Real transit traffic duplicated to shadow
- Video streaming traffic in shadow

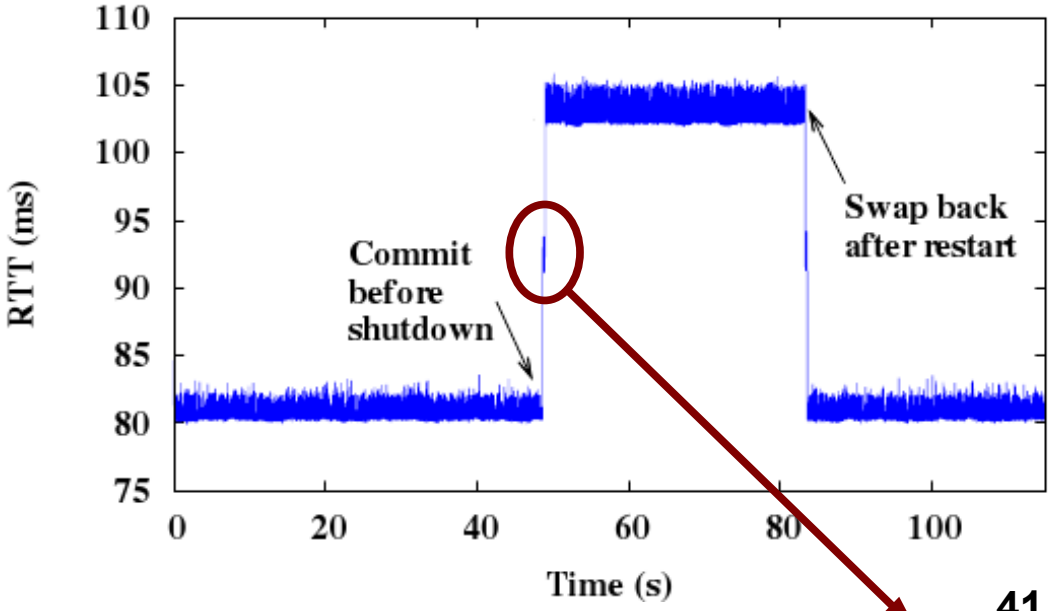
Evaluation: Router Maintenance

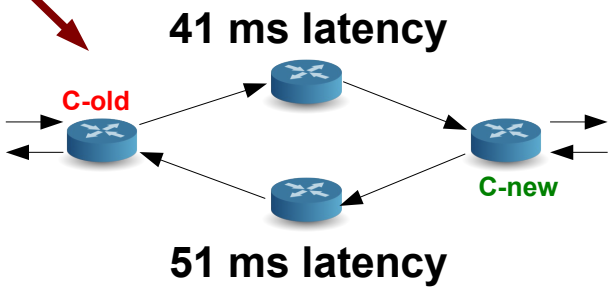


Temporarily shutdown router

- Abilene topology with 3 external peers
  - Configs translated to Quagga syntax
  - Abilene BGP dumps

Evaluation: Router Maintenance





Temporarily shutdown router

- Abilene topology with 3 external peers
  - Configs translated to Quagga syntax
  - Abilene BGP dumps