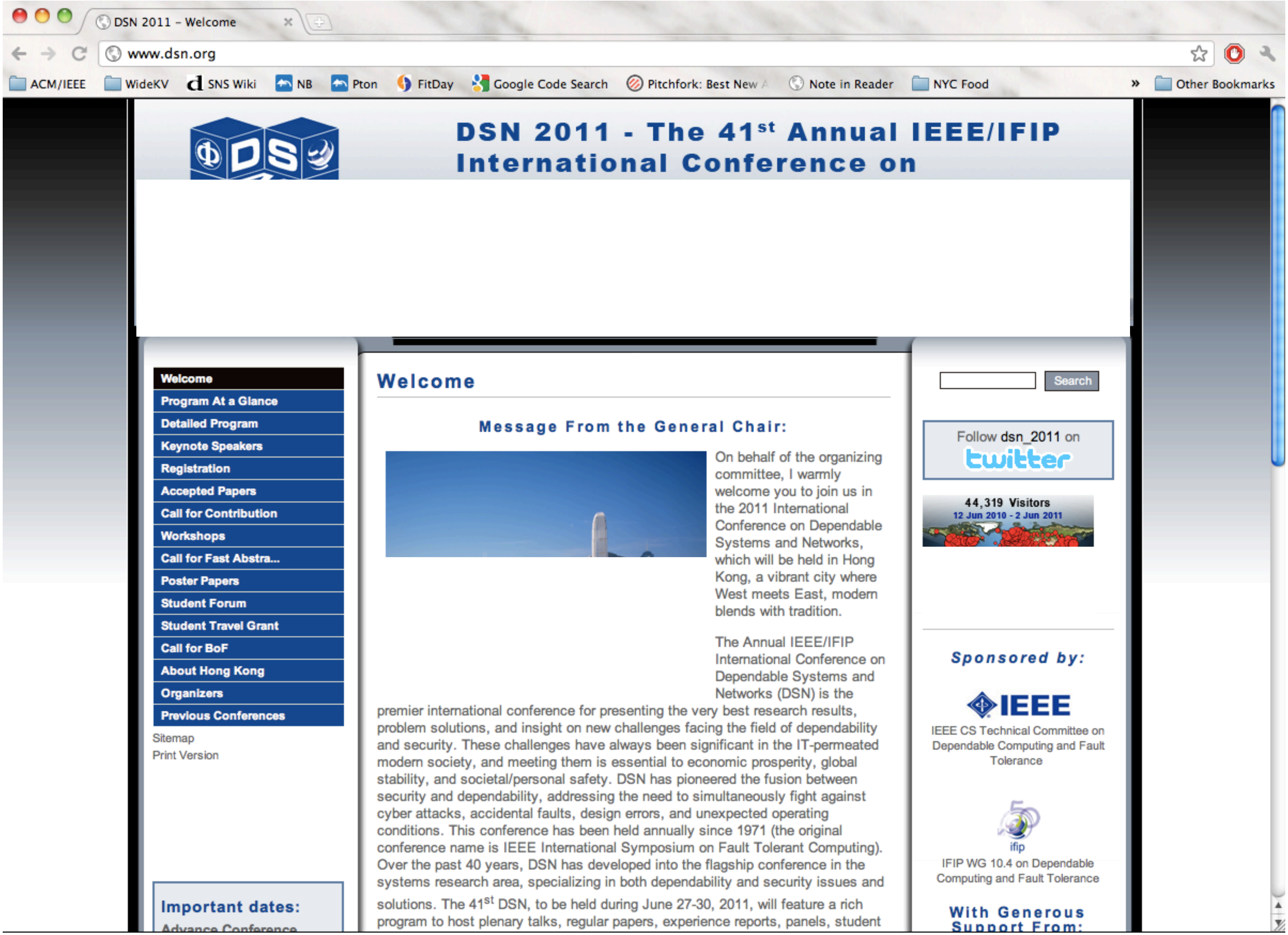
Presented at DSN-DCCS 2011 in Hong Kong on 6/28/11

TRODS

Transparent Recovery for Object Delivery Services

Wyatt Lloyd, Michael J. Freedman

Princeton University



DSN 2011 - The 41st Annual IEEE/IFIP International Conference on

Welcome

Message From the General Chair:

On behalf of the organizing committee, I warmly welcome you to join us in the 2011 International Conference on Dependable Systems and Networks, which will be held in Hong Kong, a vibrant city where West meets East, modern blends with tradition.

The Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) is the premier international conference for presenting the very best research results, problem solutions, and insight on new challenges facing the field of dependability and security. These challenges have always been significant in the IT-permeated modern society, and meeting them is essential to economic prosperity, global stability, and societal/personal safety. DSN has pioneered the fusion between security and dependability, addressing the need to simultaneously fight against cyber attacks, accidental faults, design errors, and unexpected operating conditions. This conference has been held annually since 1971 (the original conference name is IEEE International Symposium on Fault Tolerant Computing). Over the past 40 years, DSN has developed into the flagship conference in the systems research area, specializing in both dependability and security issues and solutions. The 41st DSN, to be held during June 27-30, 2011, will feature a rich program to host plenary talks, regular papers, experience reports, panels, student

Important dates:

Advance Conference

Sponsored by:

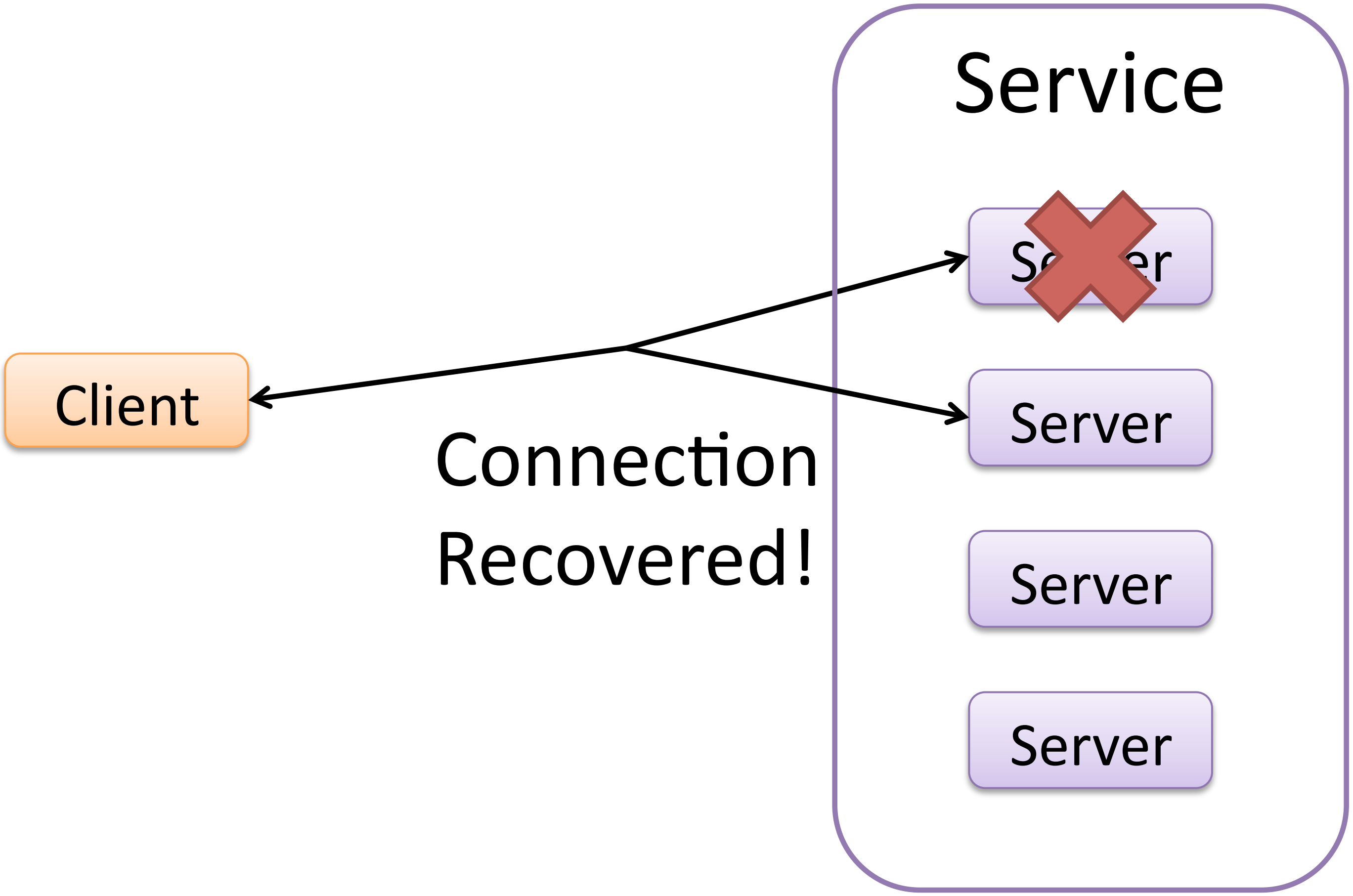
IEEE

IEEE CS Technical Committee on Dependable Computing and Fault Tolerance

ifip

IFIP WG 10.4 on Dependable Computing and Fault Tolerance

With Generous Support From:



Service

Server

Server

Server

Server

Client

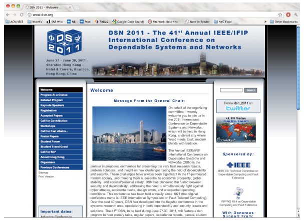
Connection Recovered!

Object Delivery Services

- Read-Only

- Static Content

- Webpages, Images, Videos



DSN 2011 - The 41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks

June 27 - June 30, 2011
Sheraton Hong Kong - Hotel & Towers, Kowloon, Hong Kong, China

Welcome

Message From the General Chair:

On behalf of the organizing committee, I warmly welcome you to join us in the 2011 International Conference on Dependable Systems and Networks, which will be held in Hong Kong, a vibrant city where West meets East, modern blends with tradition.

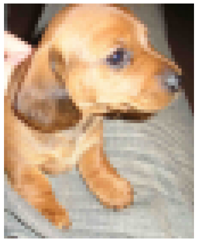
The Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) is the premier international conference for presenting the very best research results, problem solutions, and insight on new challenges facing the field of dependability and security. These challenges have always been significant in the IT-permeated modern society, and meeting them is essential to economic prosperity, global stability, and societal/personal safety. DSN has pioneered the fusion between security and dependability, addressing the need to simultaneously fight against cyber attacks, accidental faults, design errors, and unexpected operating conditions. This conference has been held annually since 1971 (the original conference name is IEEE International Symposium on Fault Tolerant Computing). Over the past 40 years, DSN has developed into the flagship conference in the systems research area, specializing in both dependability and security issues and solutions. The 41st DSN, to be held during June 27-30, 2011, will feature a rich program to host plenary talks, regular papers, experience reports, panels, student

Sponsored by:

IEEE CS Technical Committee on Dependable Computing and Fault Tolerance

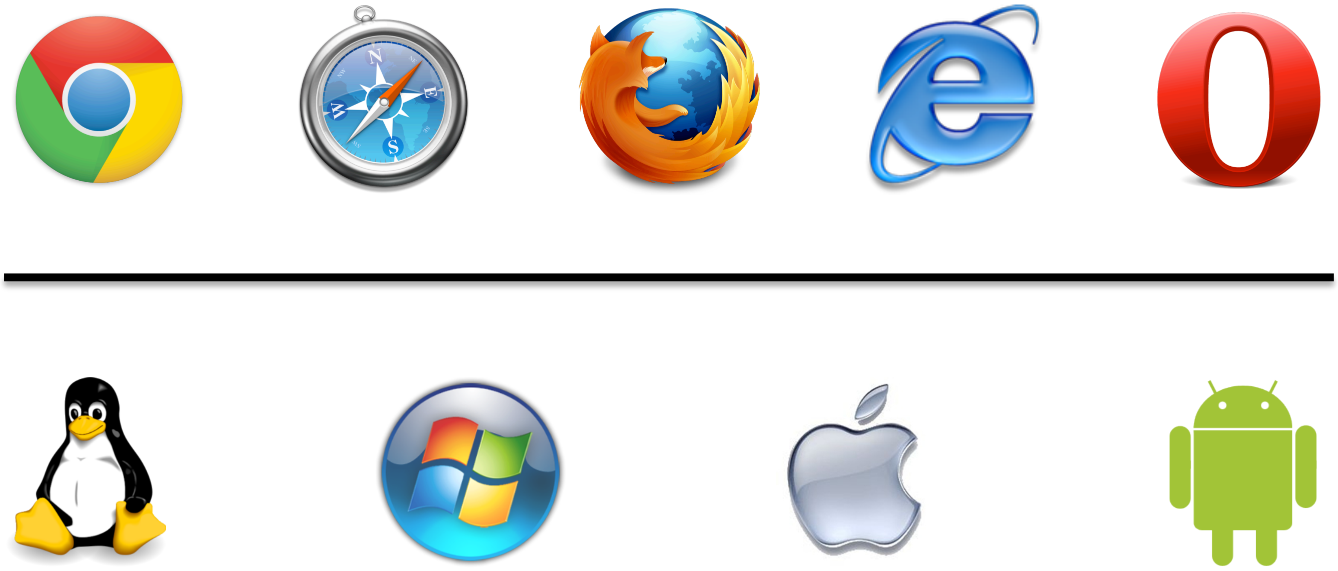
IFIP WG 10.4 on Dependable Computing and Fault Tolerance

With Generous Support From:



Work Now

- Can’t Modify Clients

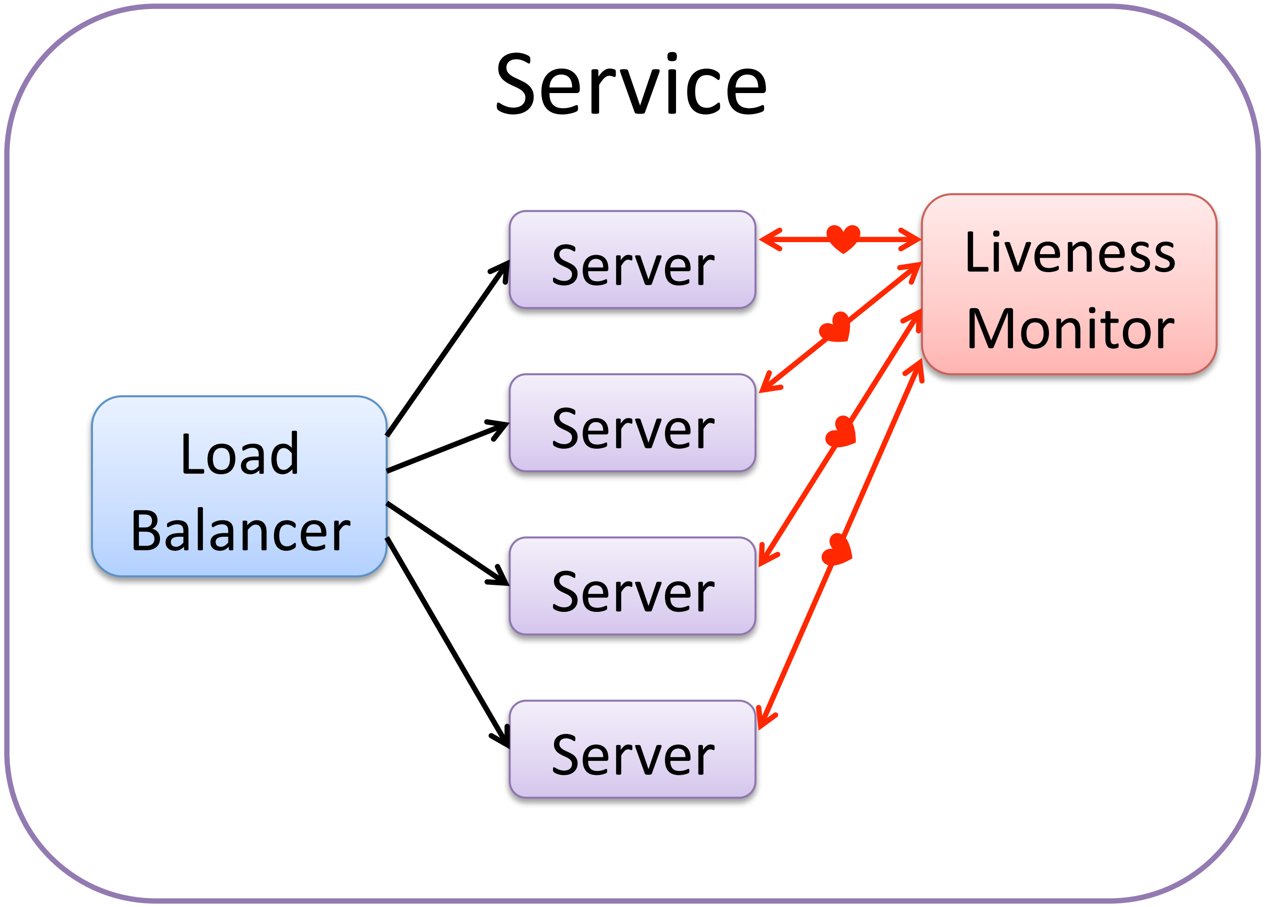


Key Idea

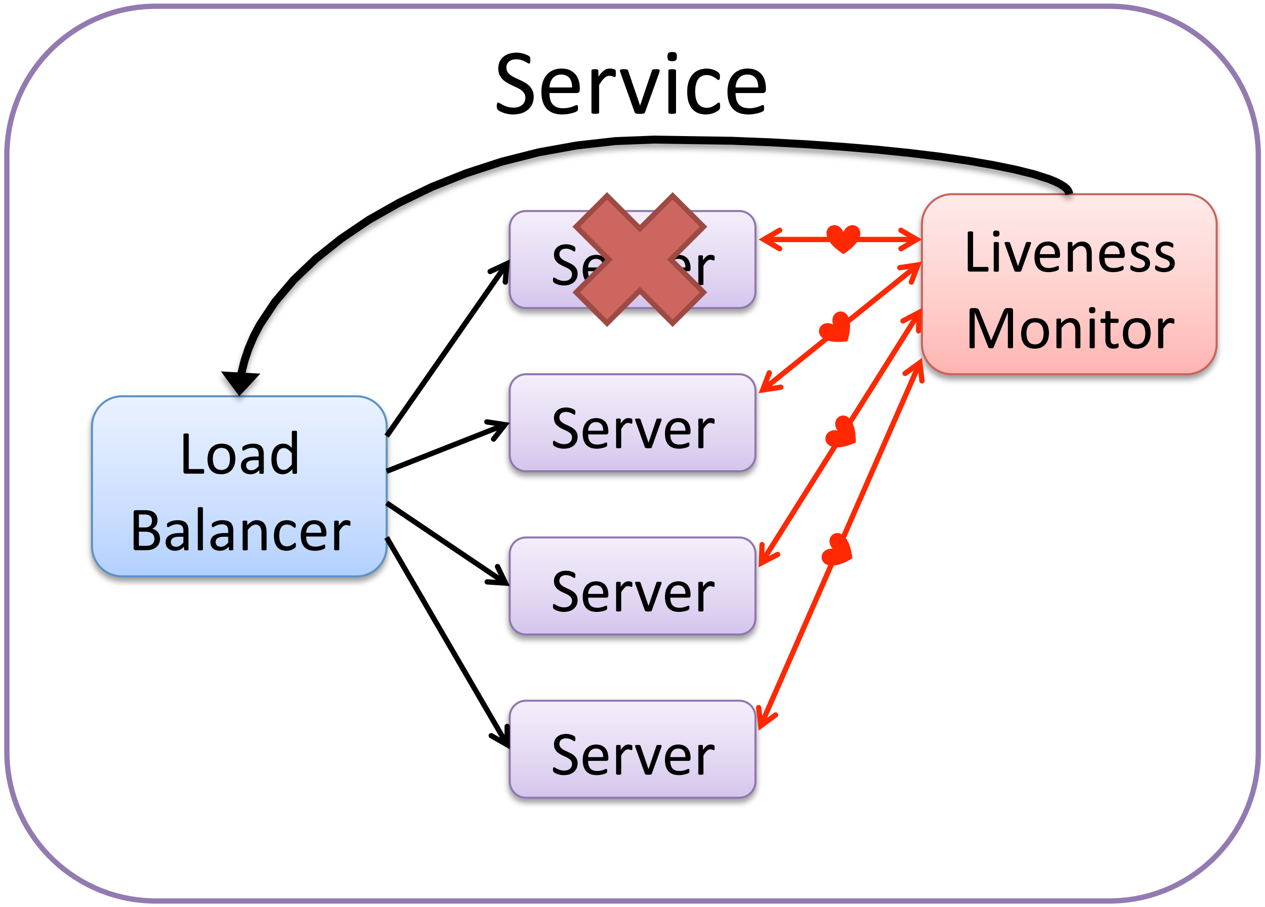
- Coerce client to help
  - To identify connections that need recovery
  - To reliably store information

- Yet client is unmodified and unaware
  - Exploit TCP spec to control client's stack

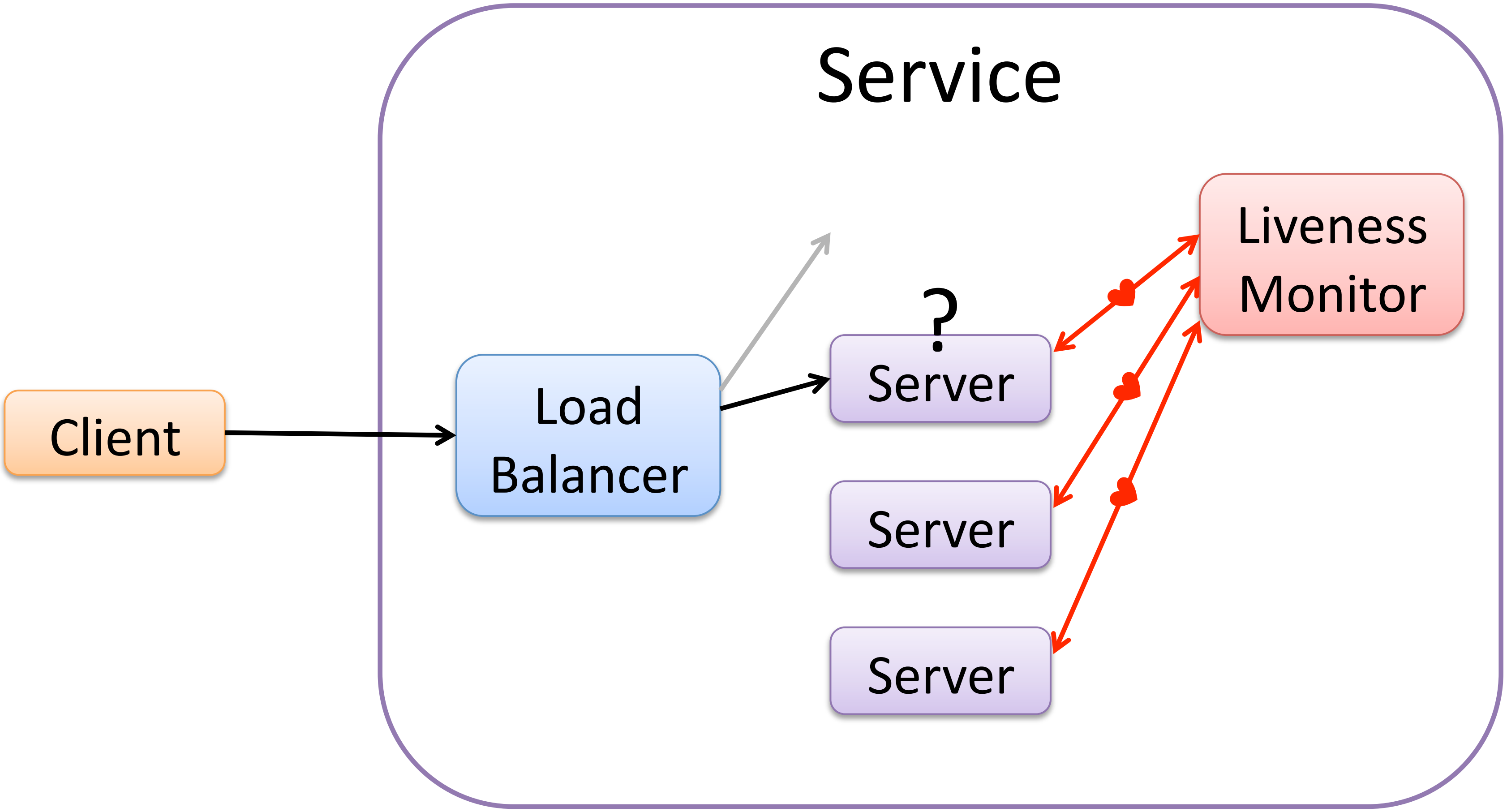
Object Delivery Cluster



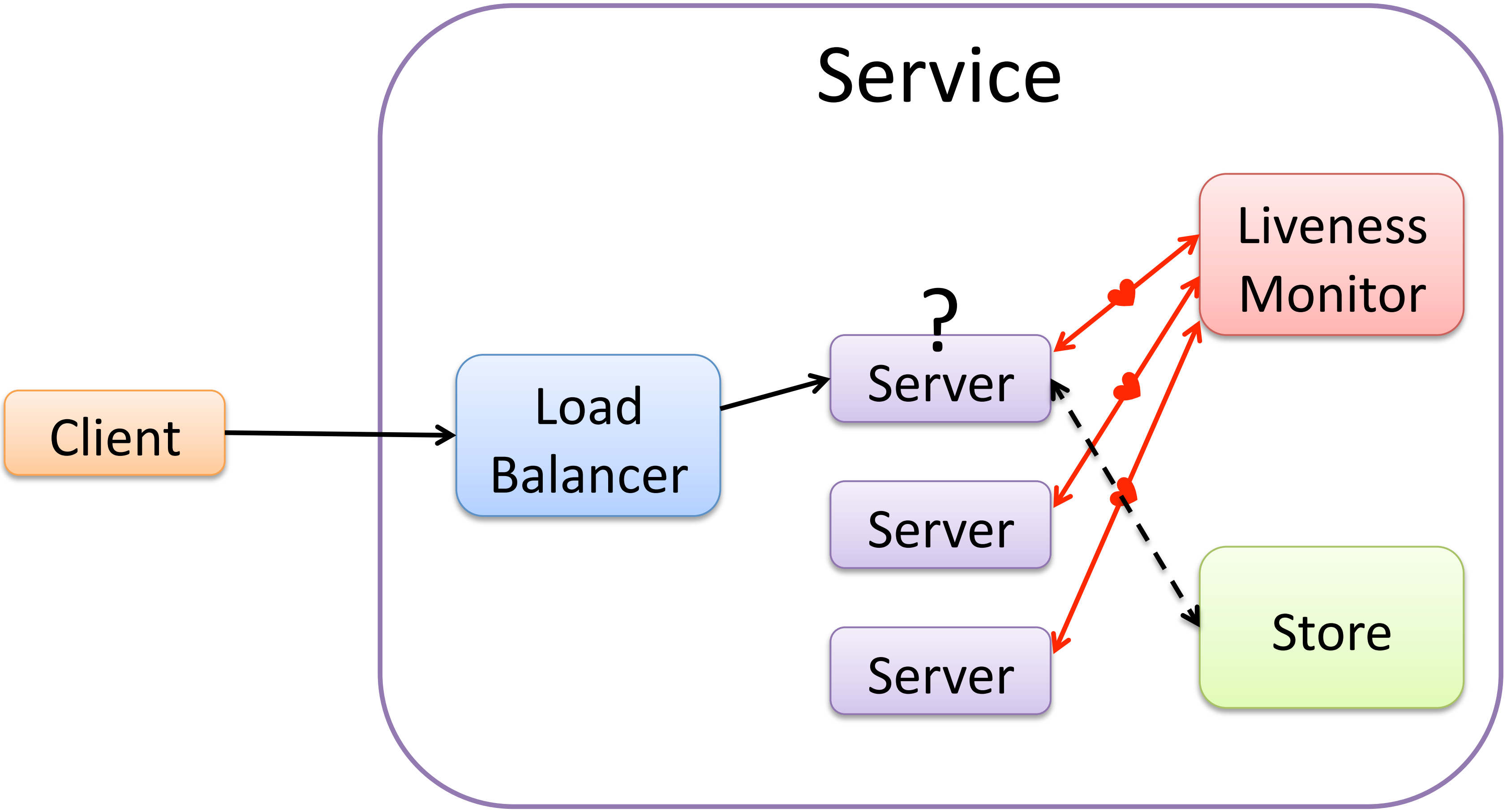
Failure


Service
Se er
Server
Server
Server
Load Balancer
Liveness Monitor

TRODS


Service
Liveness
Monitor
?
Server
Client
Load
Balancer
Server
Server

TRODS


Service
Client
Load Balancer
?
Server
Server
Server
Liveness Monitor
Store

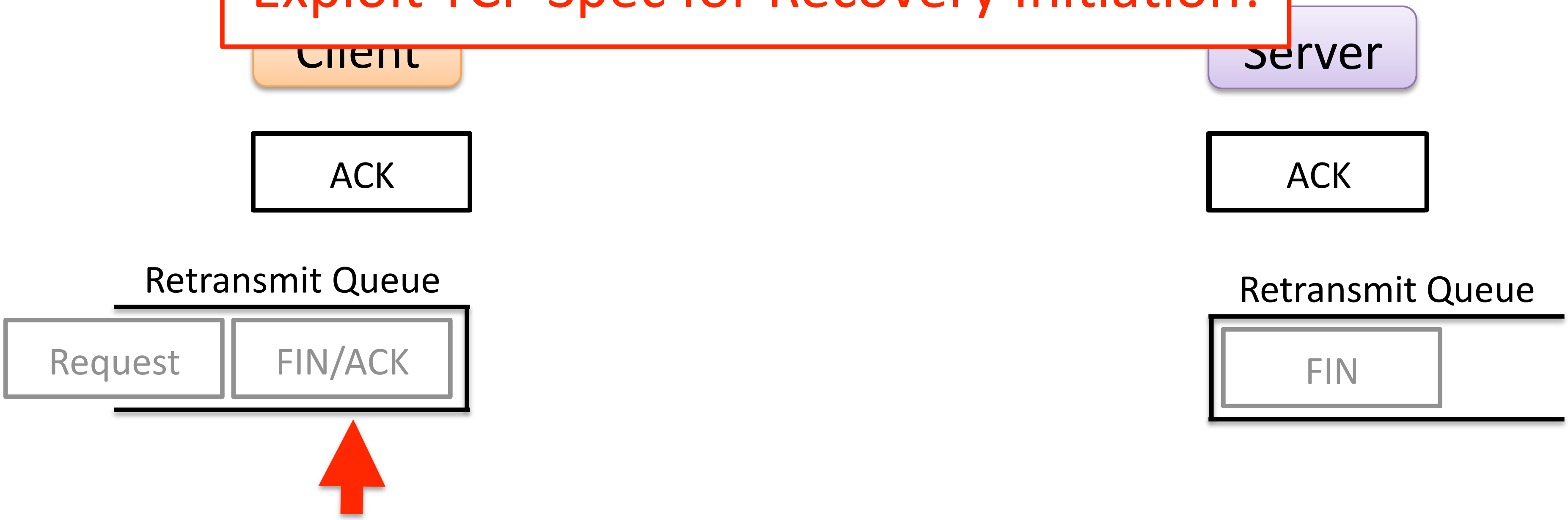
Road to Recovery

Step	**Technique**
Redirect to live server	Liveness monitor updates load balancer
New Induce client to send packet	Coerce client’s TCP stack
New Continue Connection	
Determine Phase	Use packet + stored info
Identify Object	Stored Info
Find Offset	Use packet + stored info

Coercing Clients

- Always Leave A Packet Unacknowledged

Exploit TCP Spec for Recovery Initiation!



Always Something Here

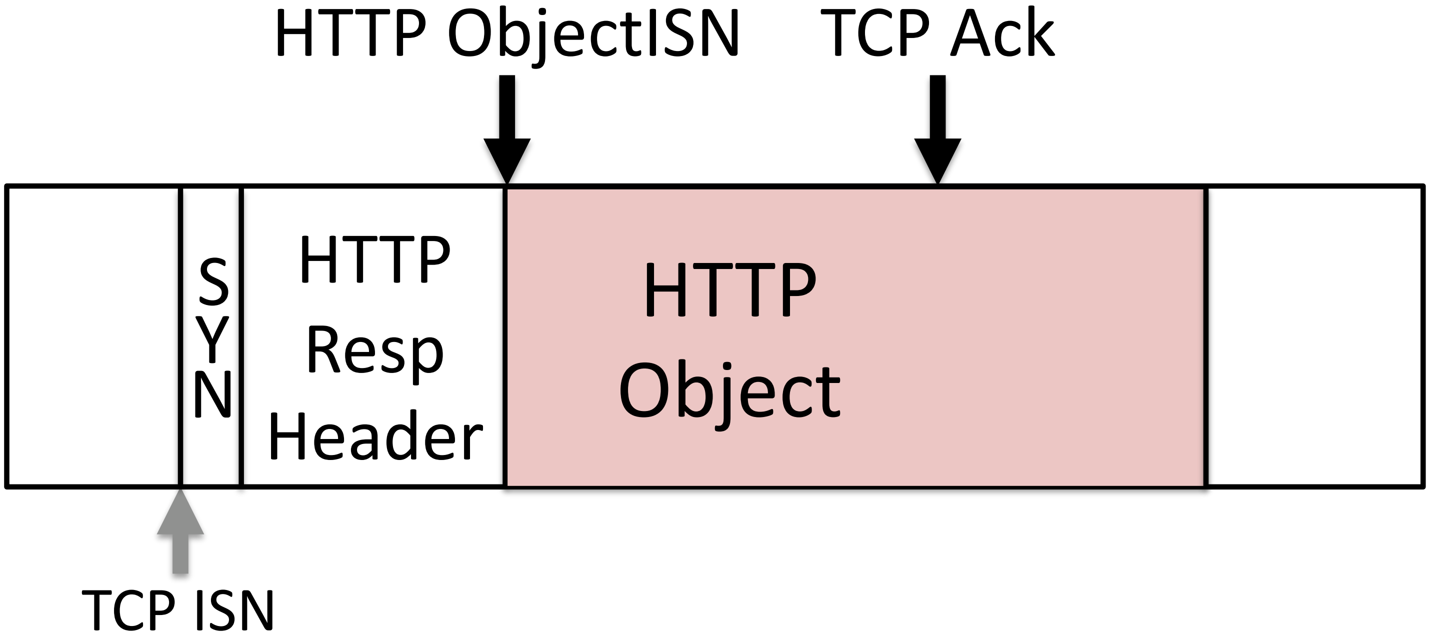
Continuing the Connection

- Determine Phase:
  1) TCP Setup
  2) HTTP Setup
  3) HTTP Download
  4) TCP Teardown

TRODS Saves Info

Continuing the Download

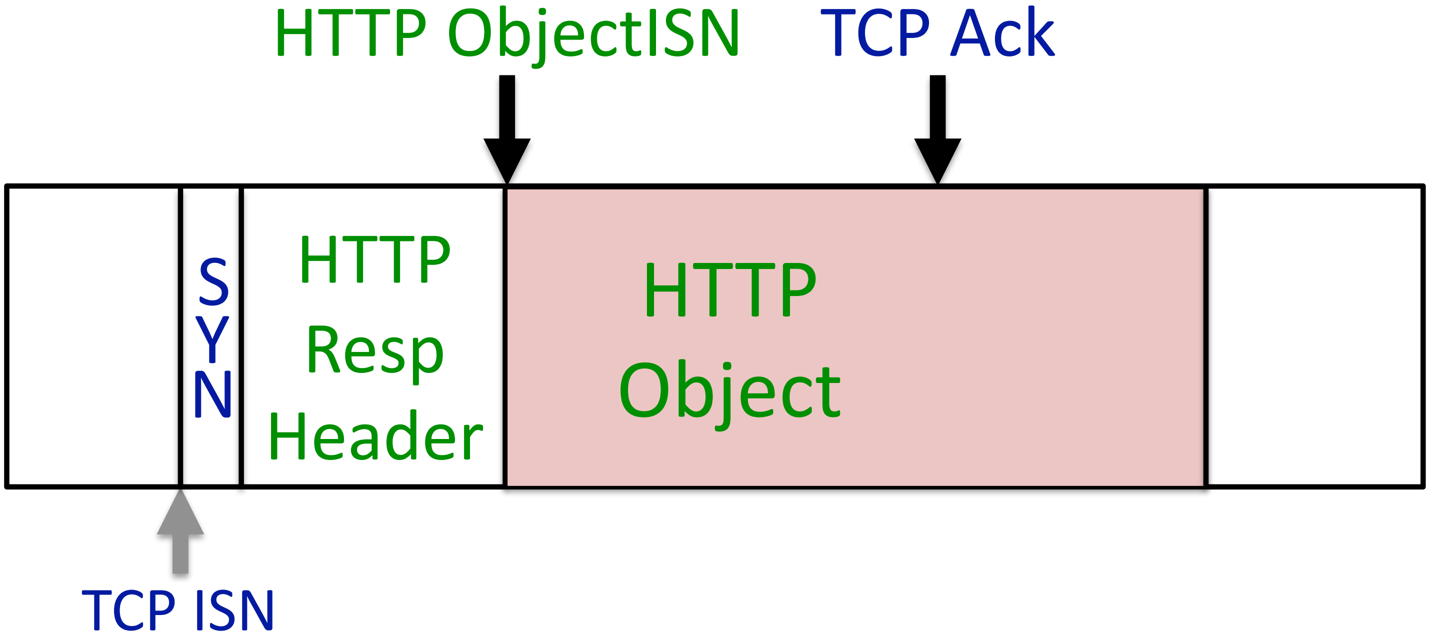
- HTTP ObjectID
- Offset = TCP Ack – HTTP ObjectISN



Continuing the Download

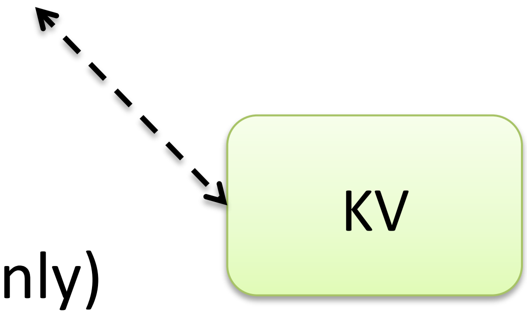
- HTTP ObjectID

- Offset = TCP Ack – HTTP ObjectISN

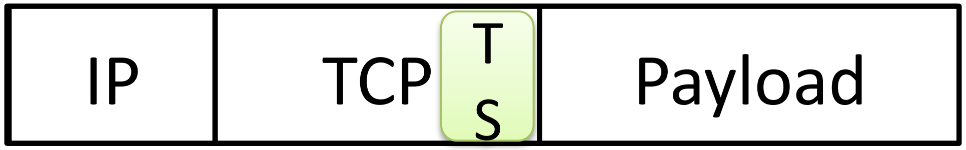


Persistent Store

- Key-Value Store
  + Corner Cases Handled
  + Unlimited Objects
  – Still Efficient (1 save only)



- TCP Timestamp
  + Very Efficient (1 machine only)
  – 1 Mill
  – Corner Cases



Exploit TCP Spec for Persistence!

Recover the Connection

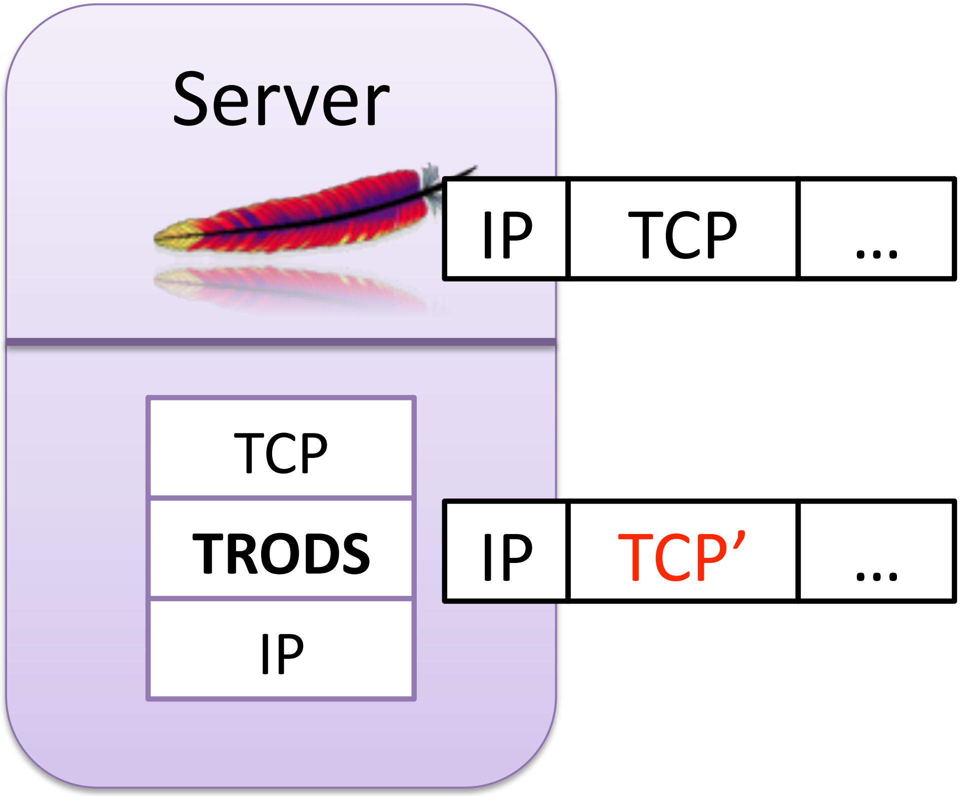
- Initiate New Connection
  - GET ObjectID …
  - Range: bytes=Offset-

- Splice Connections Together

- Works with Unmodified Servers!

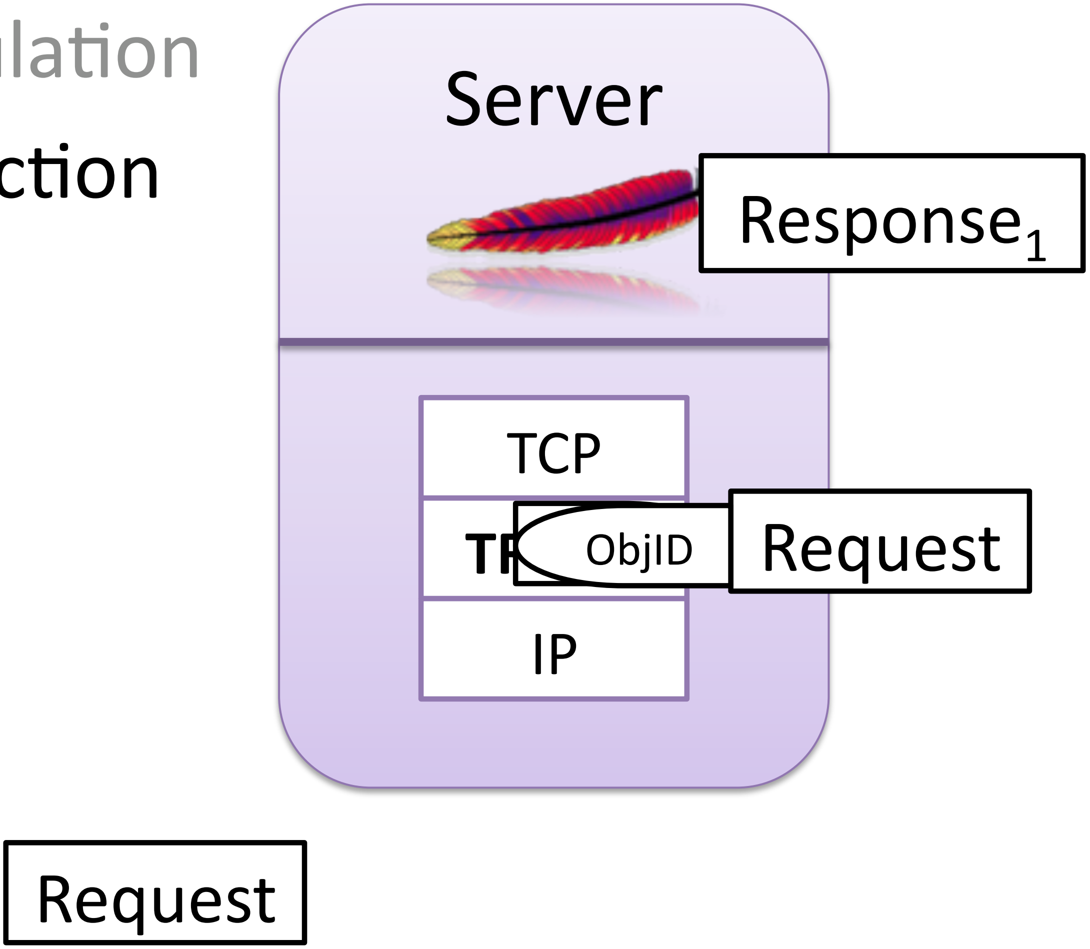
TRODS

1) Packet Manipulation



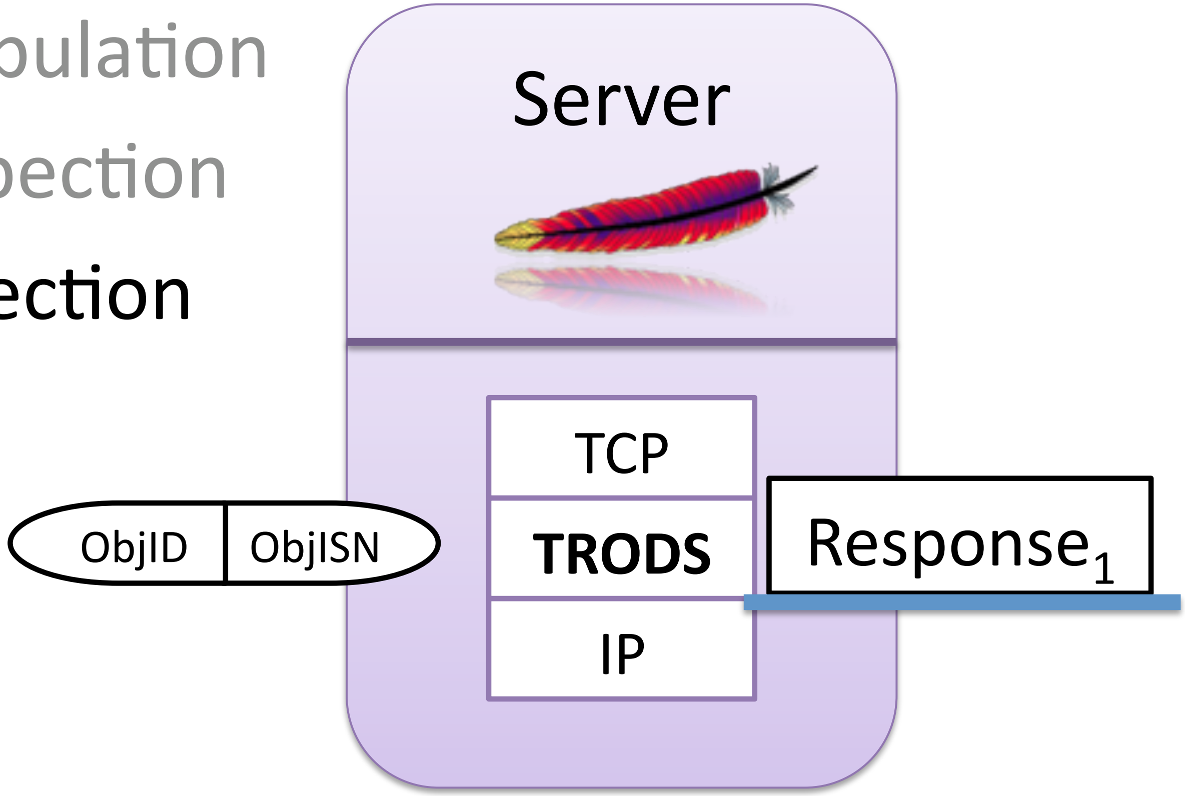
TRODS

1) Packet Manipulation
2) Protocol Inspection



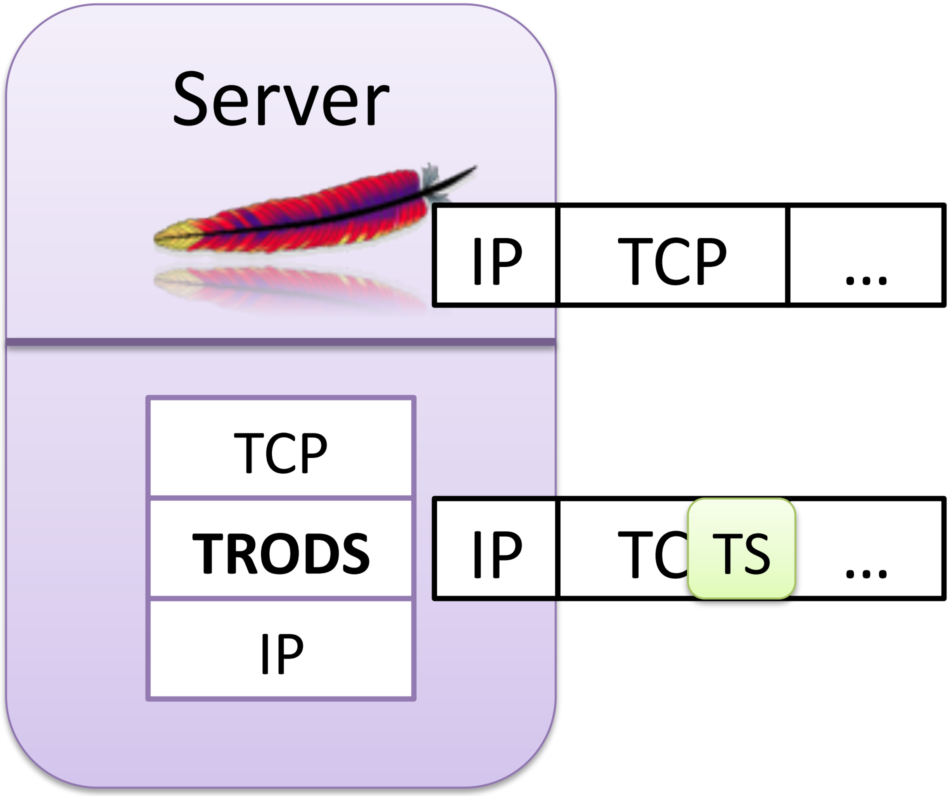
TRODS

1) Packet Manipulation
2) Protocol Inspection
3) Blocks Connection



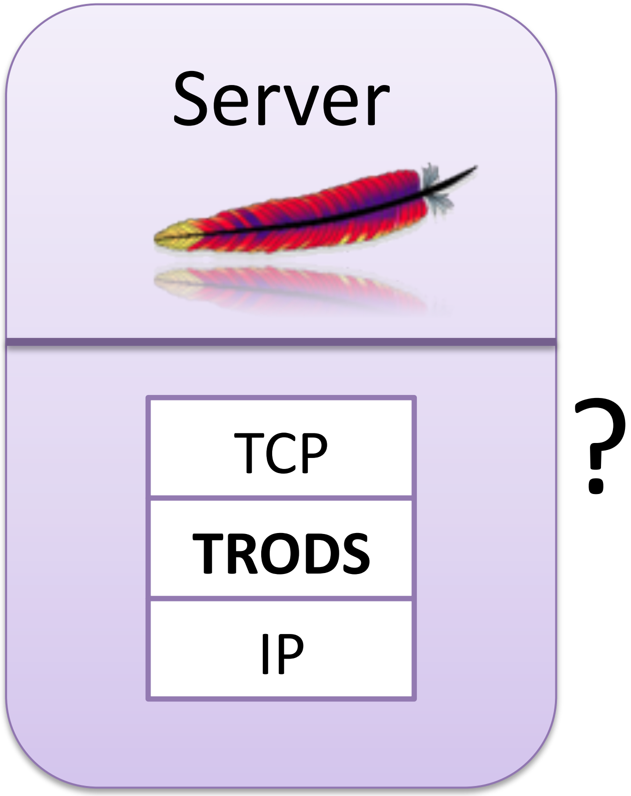
TRODS

1) Packet Manipulation
2) Protocol Inspection
3) Blocks Connection
4) State Injection



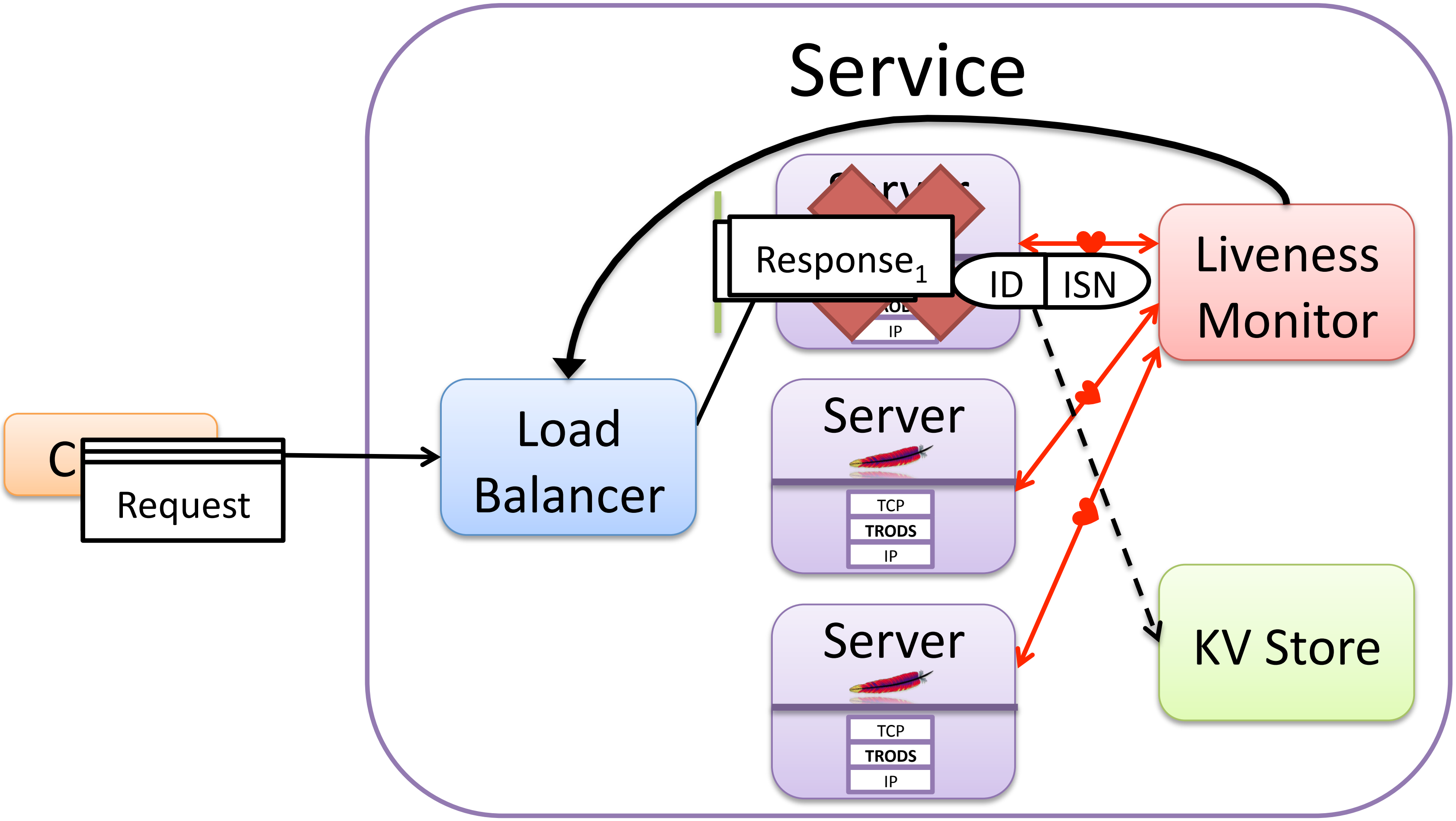
TRODS

1) Packet Manipulation
2) Protocol Inspection
3) Blocks Connection
4) State Injection
5) Recovery Initiation

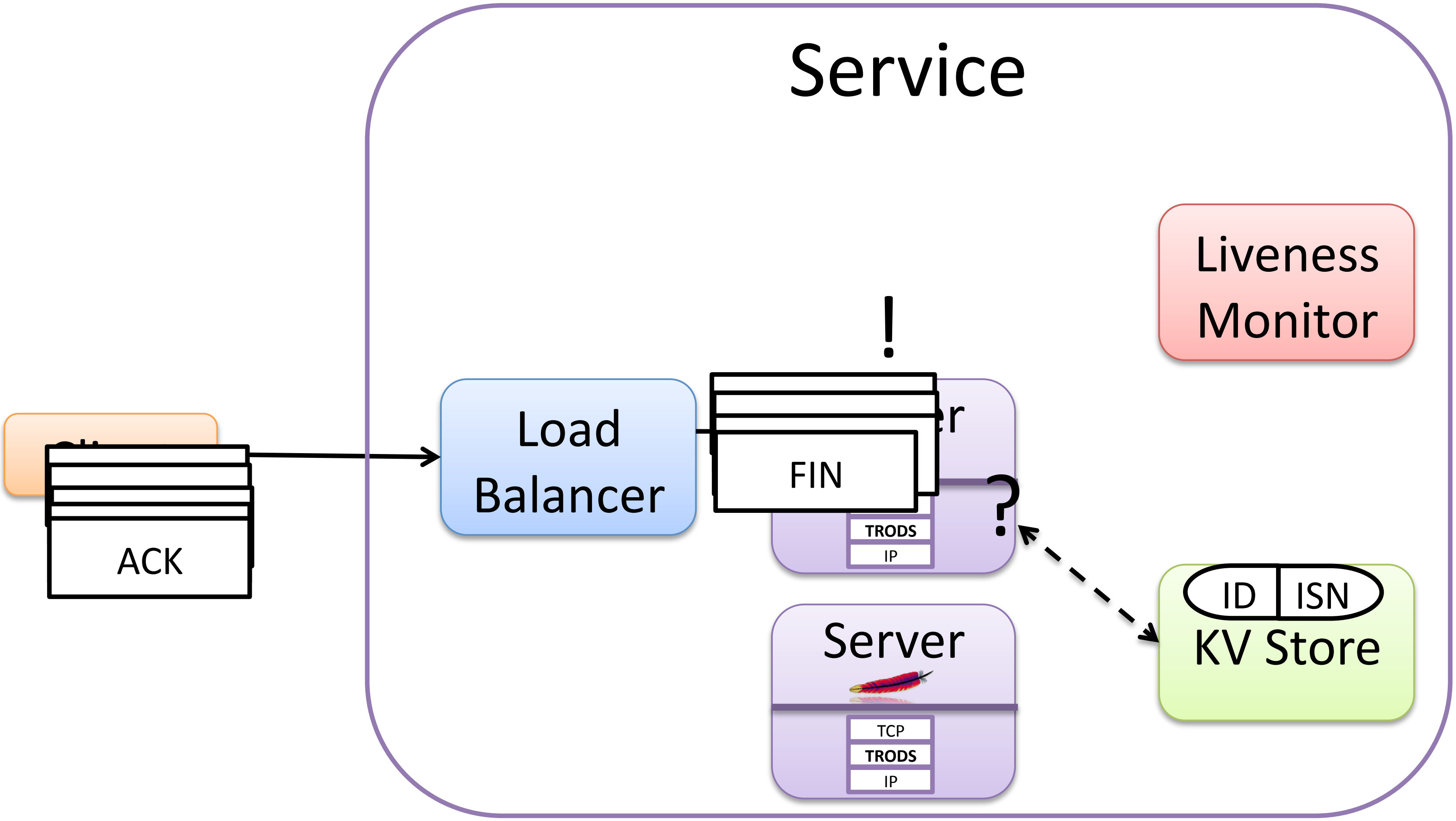


Ack

Failure Walkthrough



Failure Walkthrough



Related Work

- New Transport
  - Trickles, SCTP, TCP Migrate, …
- TCP
  - **FT-TCP**, ST-TCP, Backdoors, …
- HTTP
  - **CoRAL**, …

Implementation

- Linux Kernel Module
- 3,000 lines of C
- ~CoRAL
  - Optimistic subset of CoRAL

Experiments

- Additional Latency
  - Normal
  - Failure

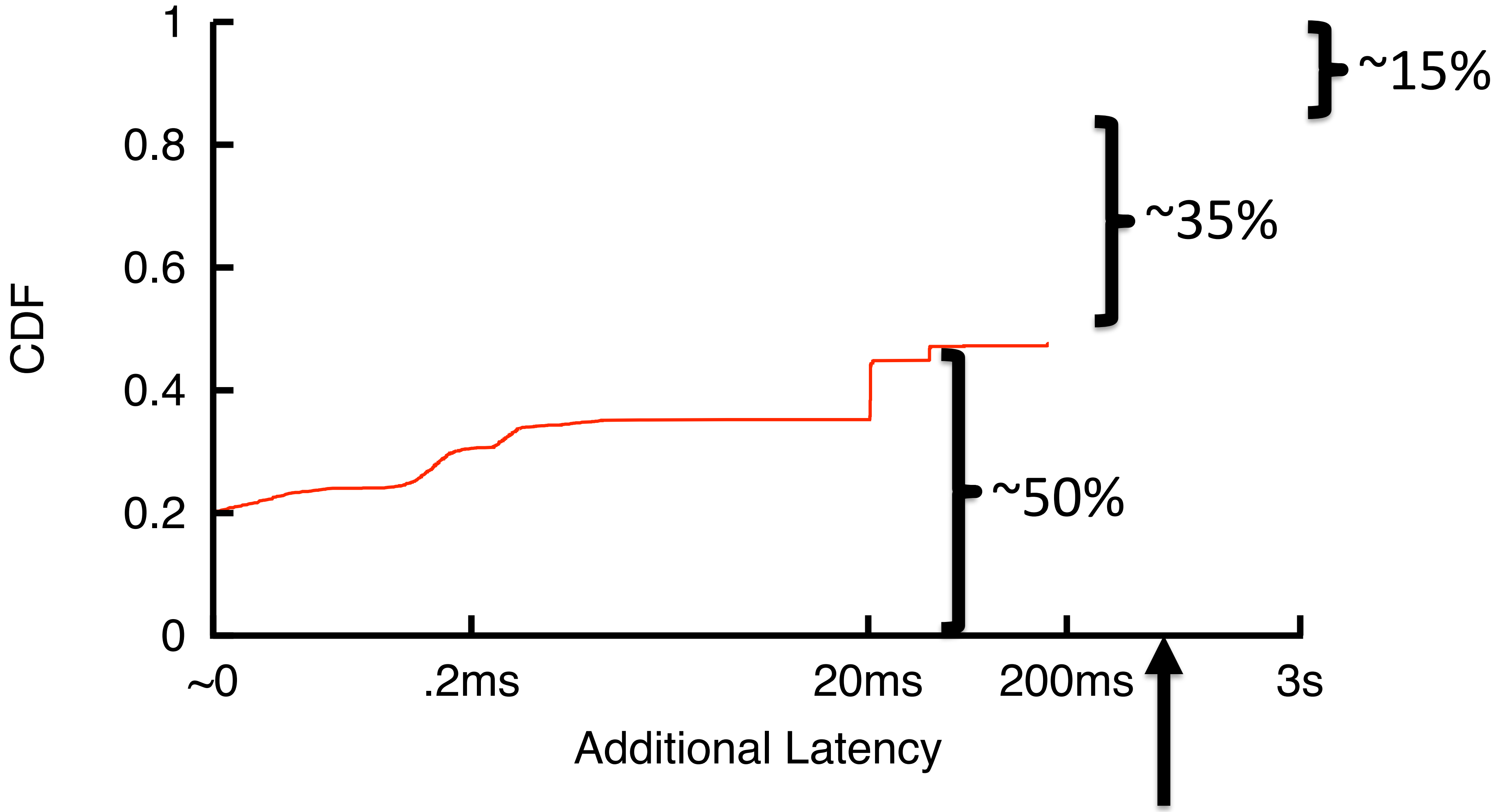
- Throughput
  - Lighttpd @ Princeton
  - Apache @ Emulab
  - Hybrid TS & KV Throughput
  - Failure

Normal Case Latency

- TRODS-TimeStamp (TS)
  - Median: + 0.009 ms
  - 99th: + 0.012 ms

- TRODS-Key-Value (KV)
  - Median: + 0.137 ms
  - 99th: + 0.148 ms

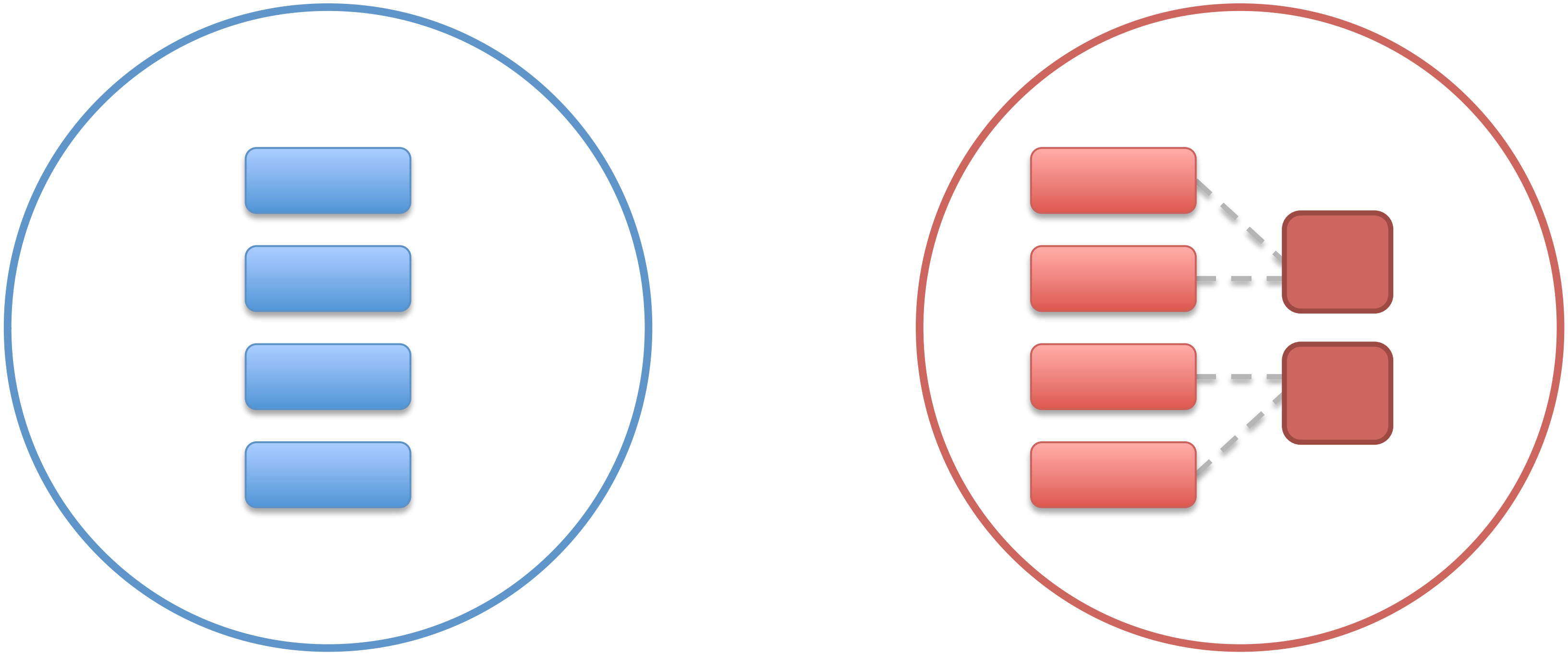
Recovery Latency



Blink of an eye

ThroughPut Per Server

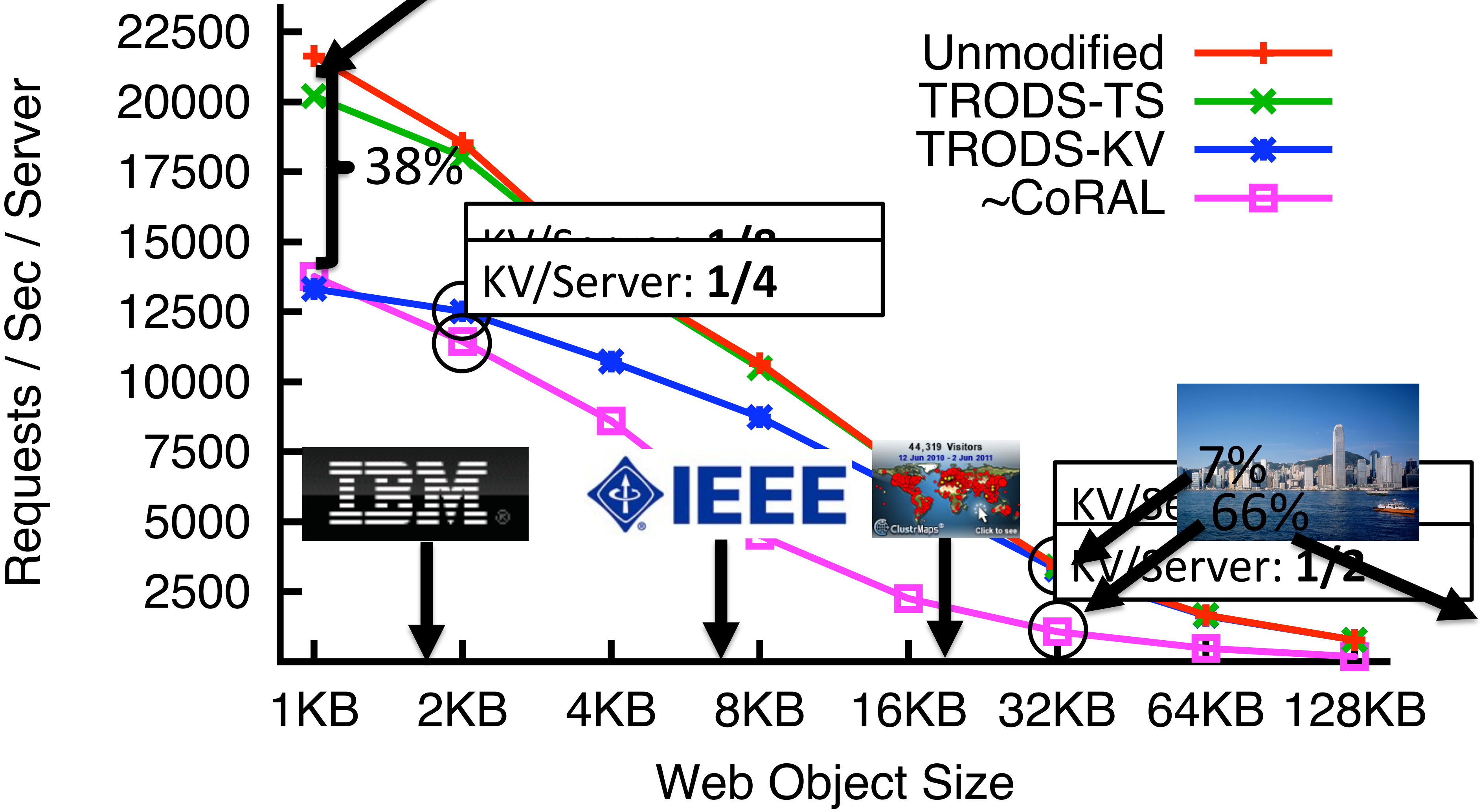
120 ops/s	Raw	120 ops/s



~~30 ops/s~~	~~Frontend~~	~~30 ops/s~~
30 ops/s/server	**TPPS**	**20 ops/s/server**

Lighttpd

9%



38%

KV/Server: **1/4**

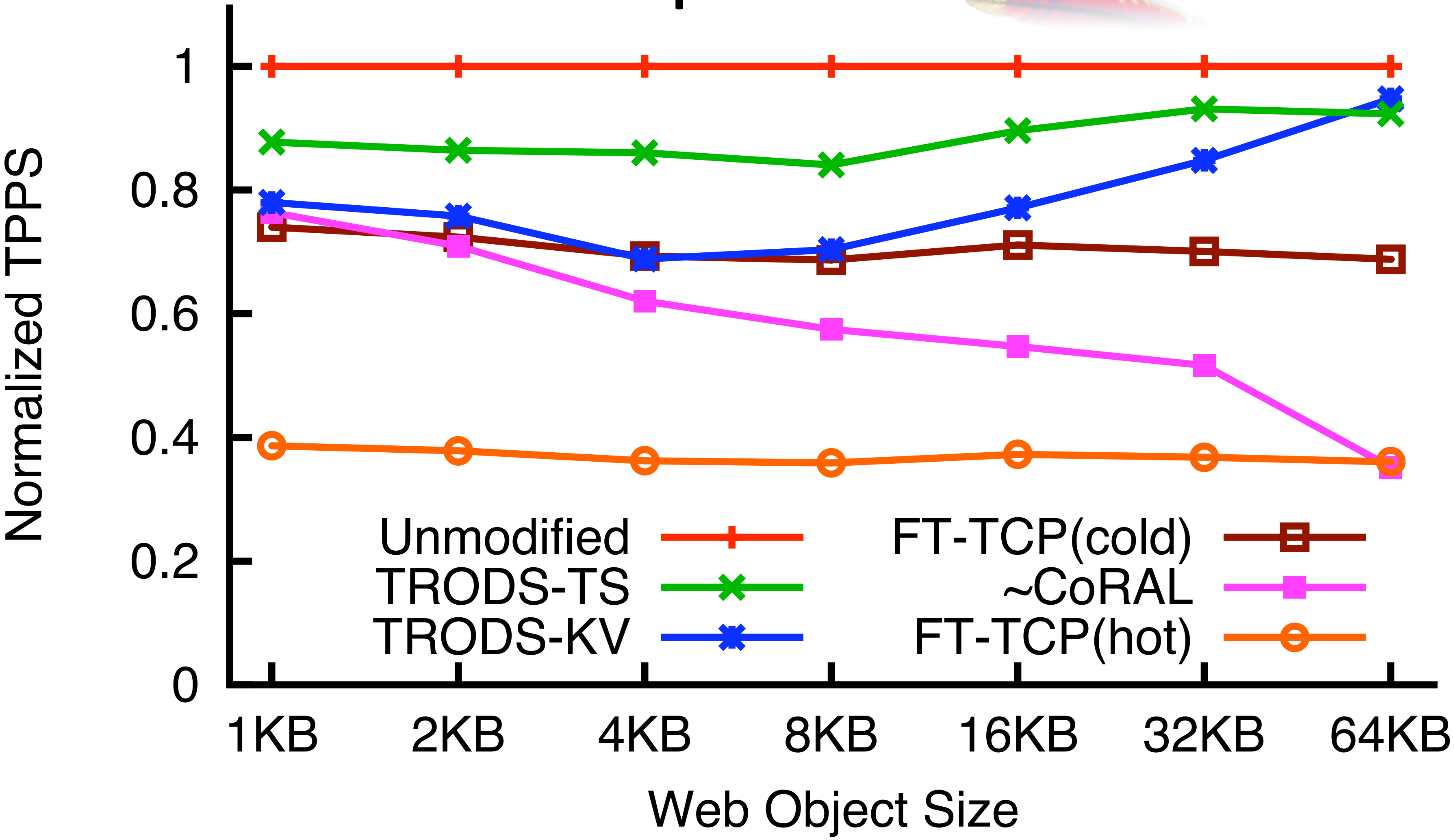
7%

66%

KV/Se

KV/Server: **1/2**

Apache


Normalized TPPS
1
0.8
0.6
0.4
0.2
0
1KB
2KB
4KB
8KB
16KB
32KB
64KB
Web Object Size
Unmodified
TRODS-TS
TRODS-KV
FT-TCP(cold)
~CoRAL
FT-TCP(hot)

Summary

- Recover Object Delivery Connections

Unmodified

- Exploit TCP Specification to Coerce ^Clients
  - To send recovery-starting packets
  - To provide persistent storage

- Evaluation
  - Low Latency
  - High Throughput Per Server

Summary

- Recover Object Delivery Connections

Unmodified
- Exploit TCP Specification to Coerce ^Clients
  - To send recovery-starting packets
  - To provide persistent storage

- Evaluation
  - Low Latency
  - High Throughput Per Server

- Questions?