



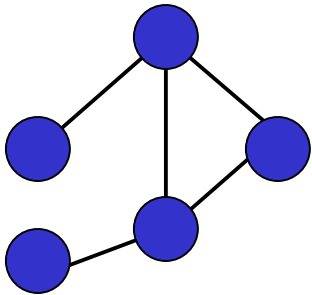
Merging Network Measurement with Data Transport

Pavlos Papageorgiou
Michael Hicks

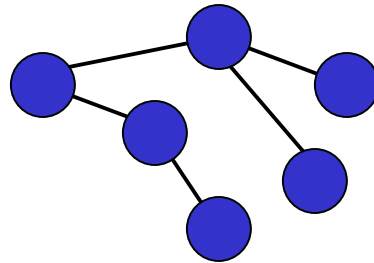
University of Maryland, College Park

Motivation

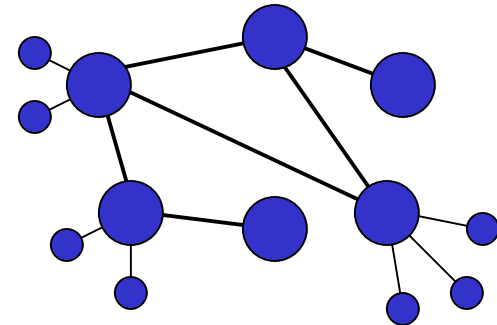
- Application-layer overlays
 - User traffic: Forward user data around
 - Probe traffic: Monitor network paths
 - Control traffic: Implement overlay algorithm



Routing Overlay



Multicast Overlay



Media Streaming Overlay

- **Everyone is measuring the network**

Prior Work

- Notion of a measurement proxy
 - Srinivasan and Zegura, M-Coop system
 - Nakao et al., Routing Underlay
- Inferring path characteristics from TCP
 - Paxson, TCP packet trace analysis
 - Mogul et al., export TCP state
 - Man et al., adjust TCP transmission intervals
- Probe delegation
 - Gonzalez and Paxson, pktd

Active Network Measurement

- Inject probes into the network and try to **infer** characteristics about the path based on what **happened** to the probes
 - Send packet pairs, packet trains
 - Use model for cross traffic and router queues
 - Interpret arrival times, inter-packet spacing
 - Elicit responses from routers, ICMP TTL
 - Requires peer entity on other end
 - Intrusive, interferes with user traffic

Passive Network Measurement

- Observe packets on the network and try to infer characteristics about the path
 - Usually at an aggregation point
 - Interpret timestamps, losses
 - Non-intrusive
 - Single interception point
 - Not flexible, cannot shape probe traffic, specify probe packet size
 - No measurements when user traffic is absent

Can we do better?

- Problem

- Active measurement tools steal bandwidth
- Passive measurement not flexible
- Trade-off between overhead and accuracy
- Can we get the best of both worlds?

- Objective

- Minimize active probing bandwidth
- Maintain same level of accuracy and flexibility
- Alternative to injecting dummy probe bits?
- Can we harness existing user data for our probes?

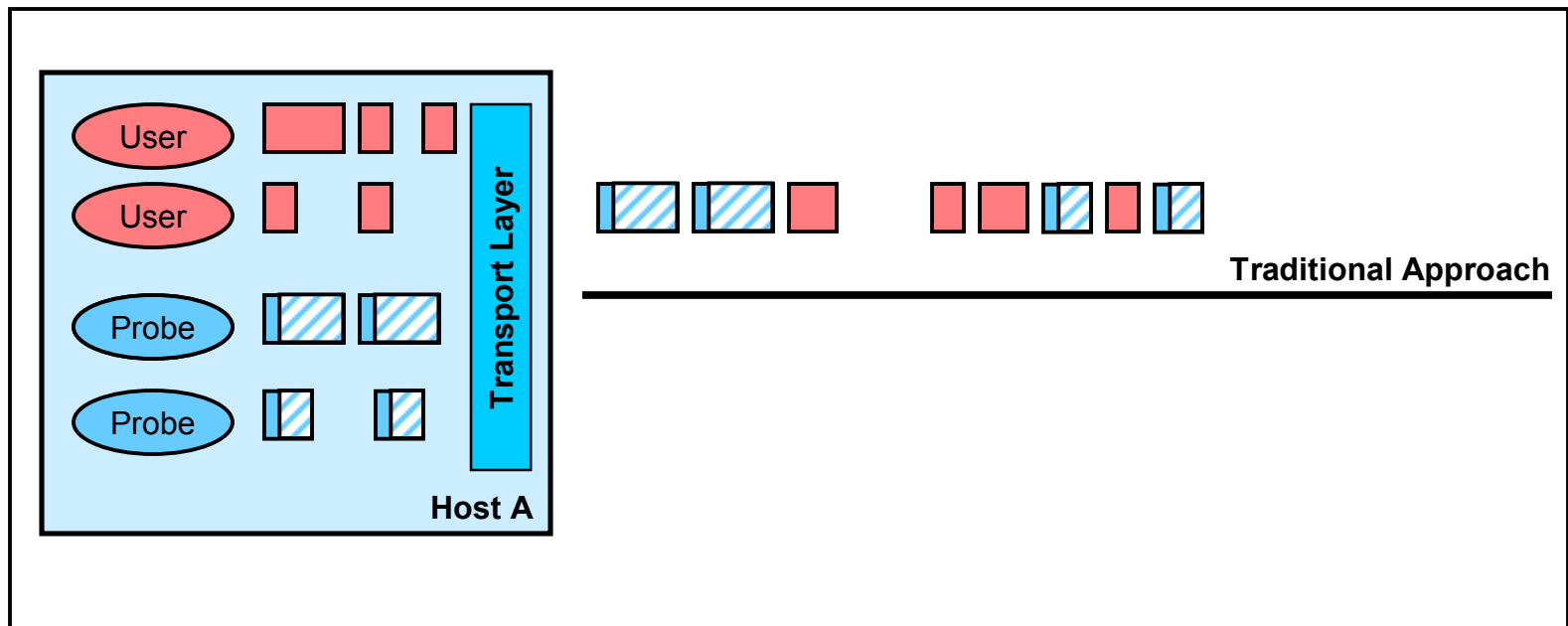
Dual nature of Data and Probe Bits

- User Data packets
 - All bits are useful at the destination
 - Not useful while packet is in transit
- Probe packets are different
 - Mostly contain empty padding
 - Useful only while **passing** through the network
 - Discarded at the destination

We can reuse the empty padding bits
to carry useful user data bits

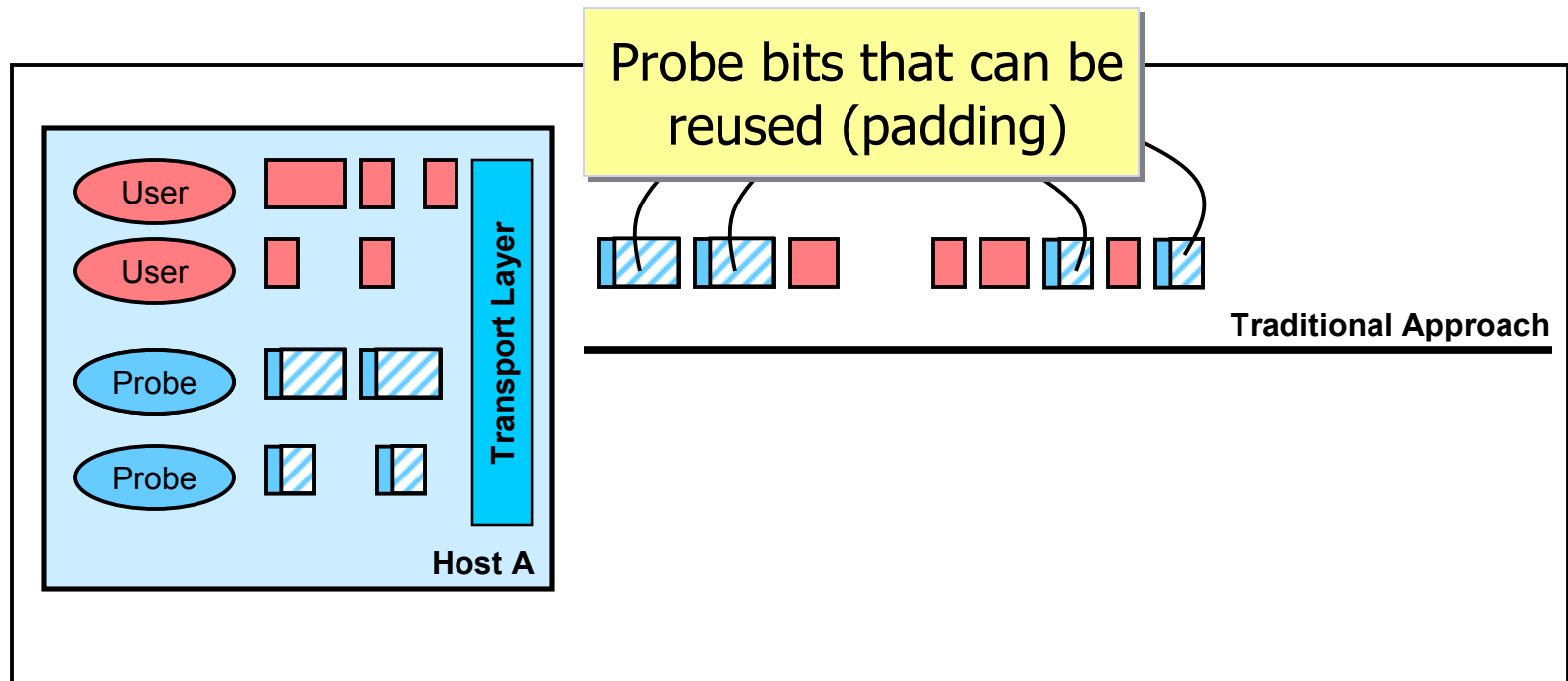
Measurement Aware Transport

- Take advantage of dual nature of the bits
- Reuse the probe padding to get more out of every bit transported in the network



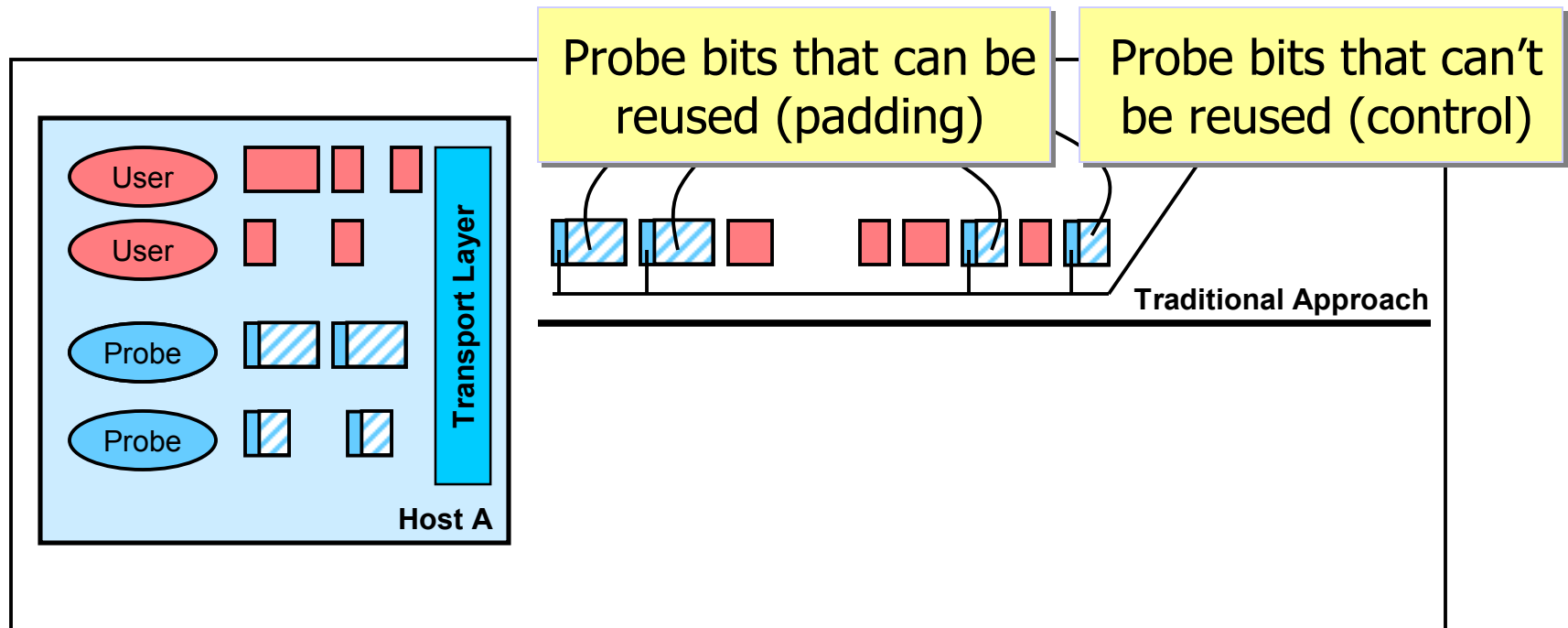
Measurement Aware Transport

- Take advantage of dual nature of the bits
- Reuse the probe padding to get more out of every bit transported in the network



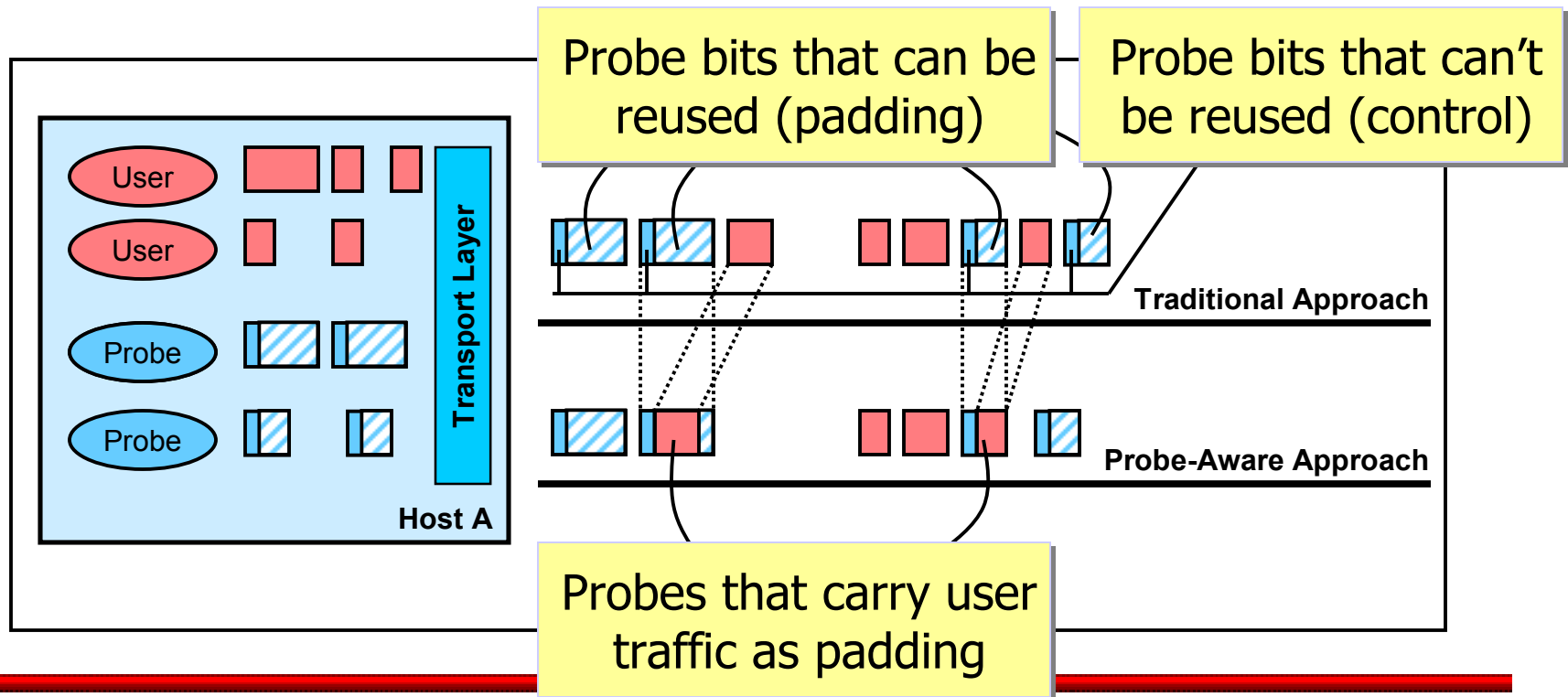
Measurement Aware Transport

- Take advantage of dual nature of the bits
- Reuse the probe padding to get more out of every bit transported in the network



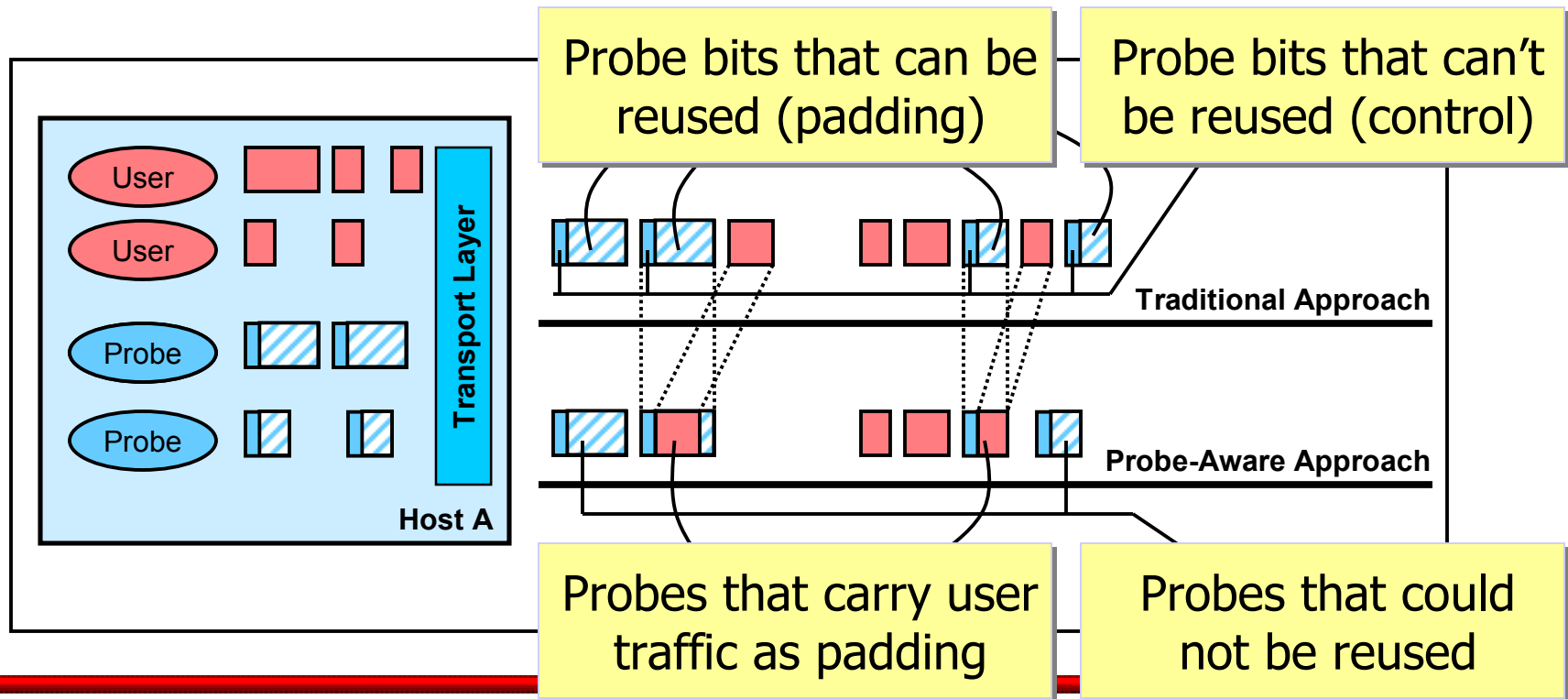
Measurement Aware Transport

- Take advantage of dual nature of the bits
- Reuse the probe padding to get more out of every bit transported in the network



Measurement Aware Transport

- Take advantage of dual nature of the bits
- Reuse the probe padding to get more out of every bit transported in the network



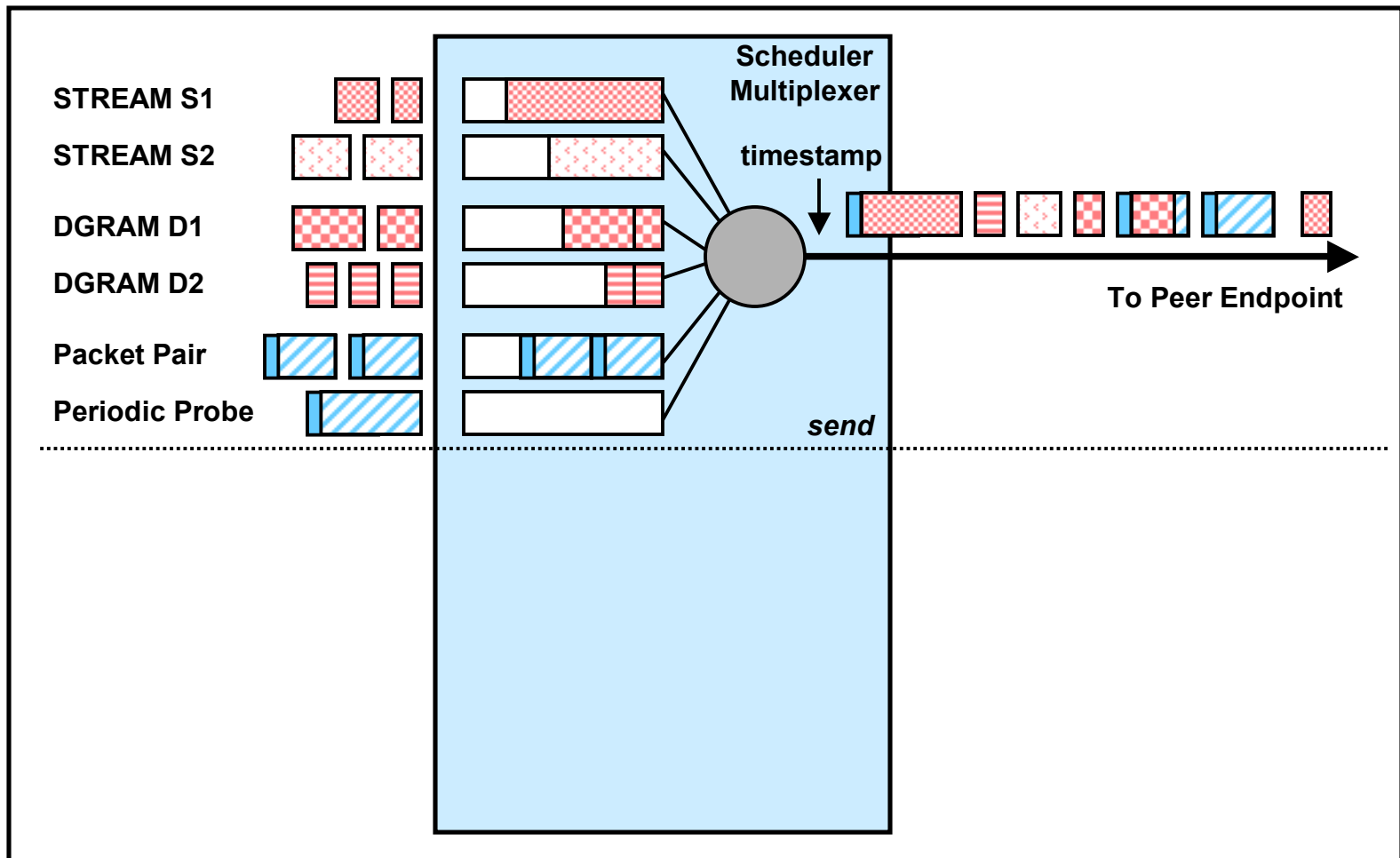
Implementation

- Probe-aware transport agent prototype
 - Agent must run on each IP endpoint host
 - Linux implementation in user space
 - Acts as regular transport layer
 - Exports expanded BSD socket interface
 - Applications that do not send probes:
 - No changes needed – no need to recompile
 - All socket calls intercepted by preloaded library
 - Measurement tools that send probes:
 - Need to specify padding portion of probes

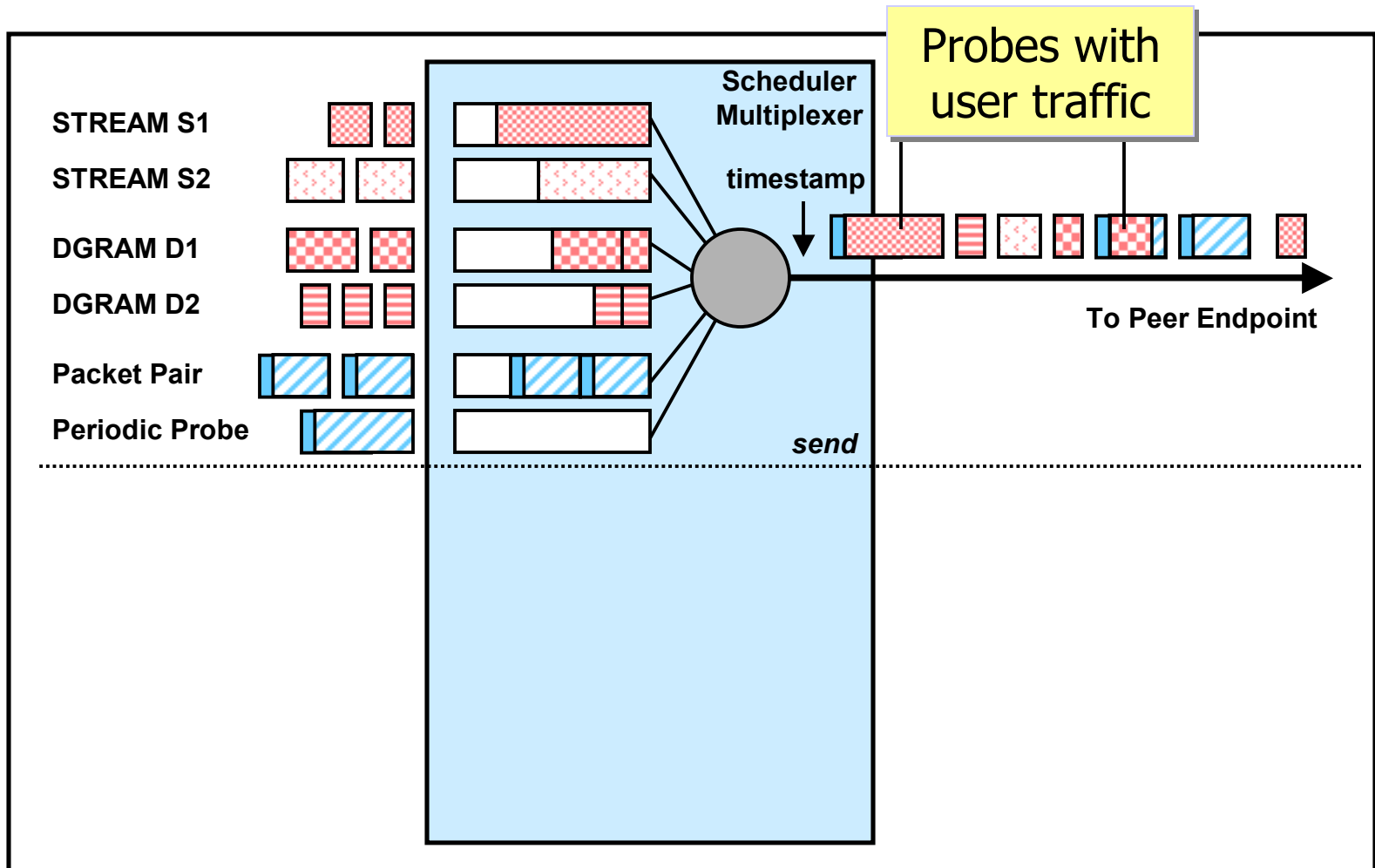
Implementation - Internals

- All flows **multiplexed** into a single stream
 - TCP and UDP (demultiplexed at receiver)
- **Unified** congestion control across all flows
 - All TCP flows
 - Any UDP flows that use the enhanced API
- Schedule for maximal **reuse of padding**
 - API for marking UDP packets as probes
 - Probes have time and packet size constraints
 - Probe padding is filled with TCP or UDP data

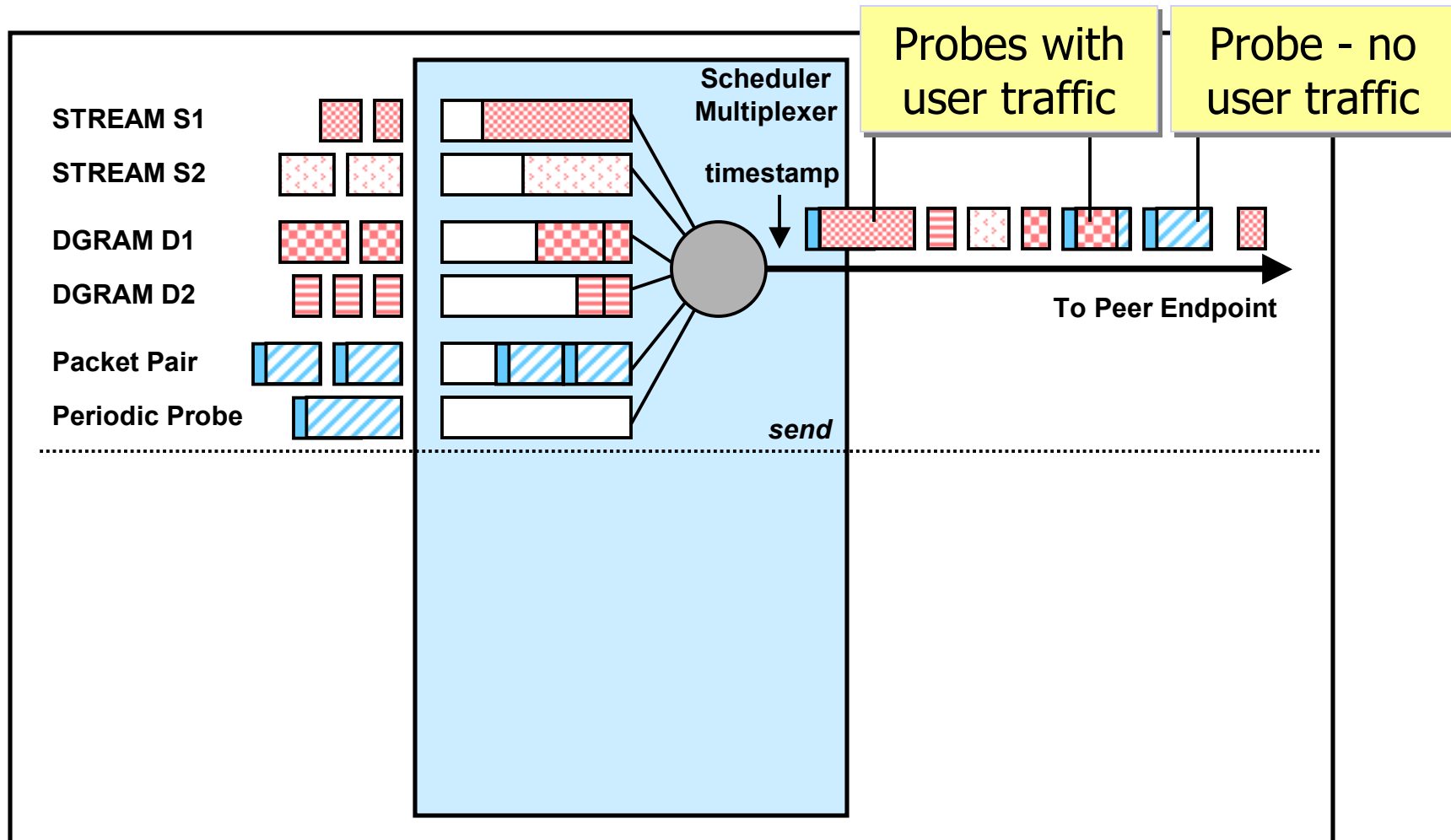
Probe Aware Transport Agent



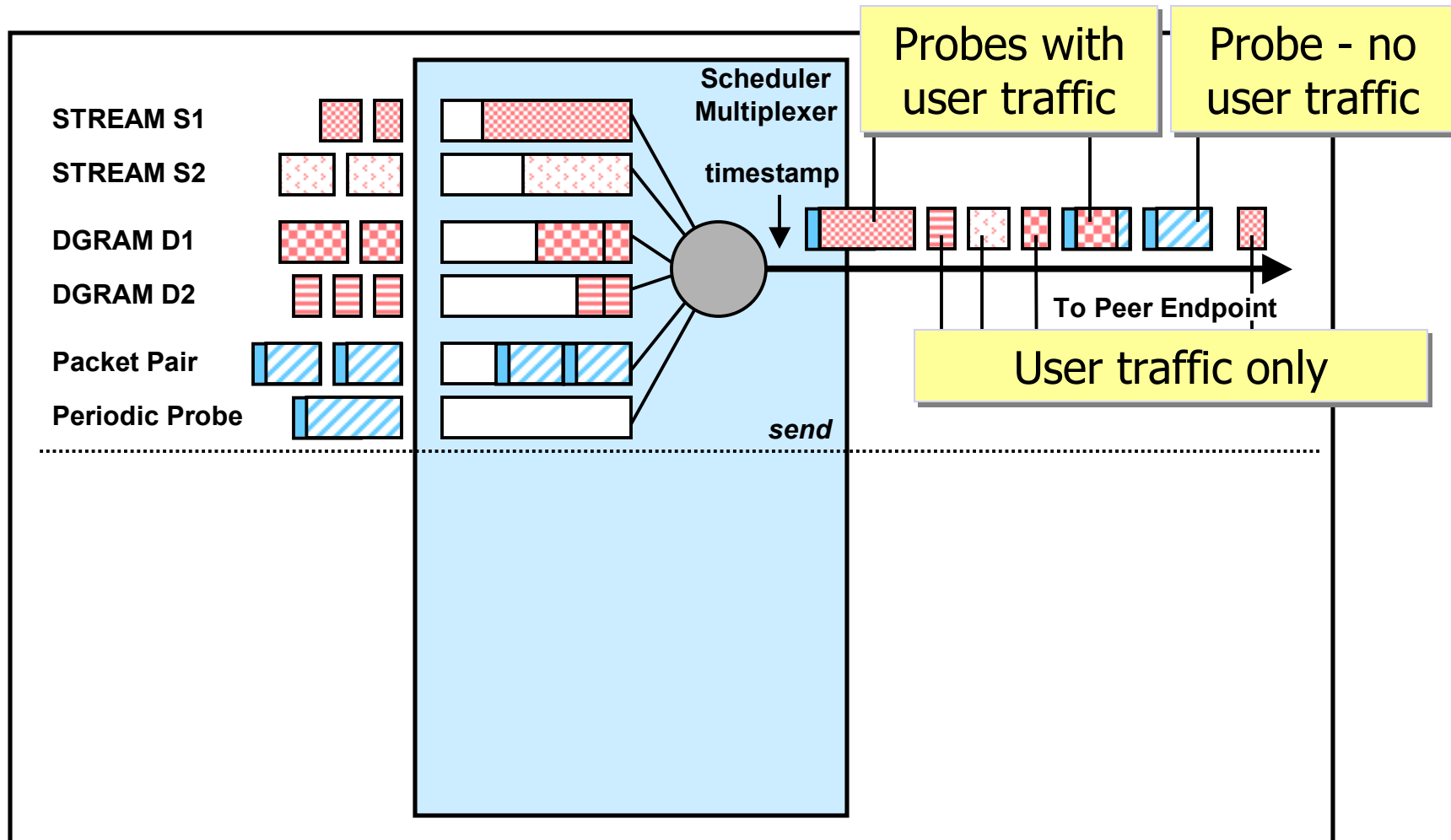
Probe Aware Transport Agent



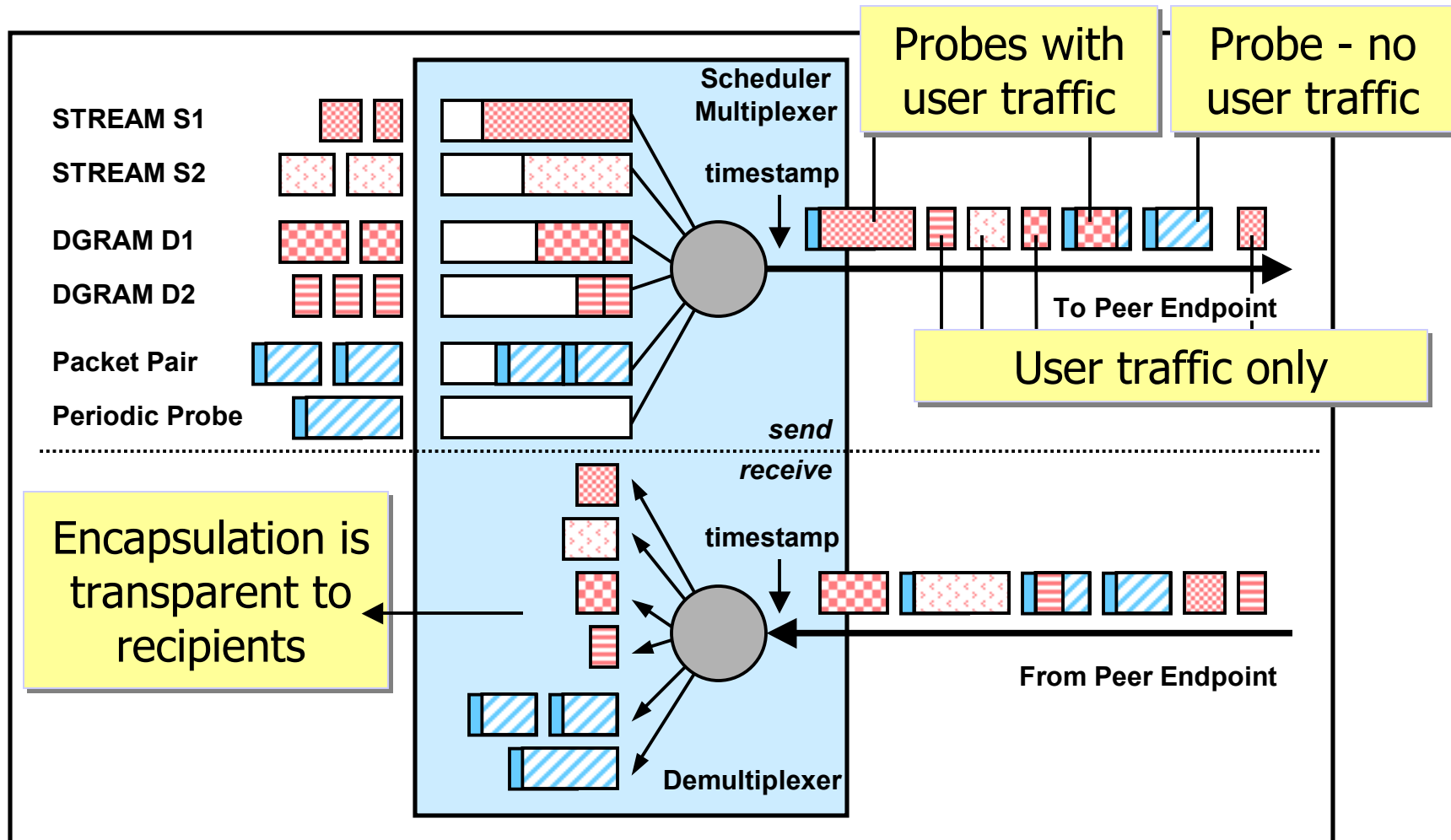
Probe Aware Transport Agent



Probe Aware Transport Agent



Probe Aware Transport Agent



Extended BSD API

- Extends BSD socket API
 - Uses ancillary data in **sendmsg** and **recvmsg**
 - For each UDP probe packet:
 - Specify which portion can be reused (padding)
 - Specify delay constraints

sendmsg() – per packet flags	
PAD_PKT [<i>s</i>]	Needs <i>s</i> bytes of padding
DELAY [<i>t</i>]	Can be delayed up to <i>t</i> msec
PKT_TRAIN	Part of packet train
sendmsg() – per session flags	
SINGLE_PKT	Do not encapsulate packets
CON_CONTROL	Under congestion control

API – Code Snippet

```
/* send a packet train of length TRAIN_LENGTH; iov buffer points to &seqno */

cmsg = CMSG_FIRSTHDR(&msg);
cmsg->cmsg_level = TPROTO_TAGENT;
cmsg->cmsg_type = TDGRAM_CONTROL_PKT;
cmsg->cmsg_len = CMSG_LEN(sizeof(tdgram_control_pkt));
ctrl_pkt = (tdgram_control_pkt *)CMSG_DATA(cmsg);

for (i = 0; i < TRAIN_LENGTH; i++) {
    /* packet contains a sequence number and the rest is padding */
    ctrl_pkt->pad_pkt = 1000 - sizeof(seqno);

    /* indicate the position within the packet train */
    ctrl_pkt->pkt_train = (i == 0) ? TPROTO_TRAIN_FIRST :
                          (i == TRAIN_LENGTH-1) ? TPROTO_TRAIN_LAST :
                          TPROTO_TRAIN_MIDDLE;

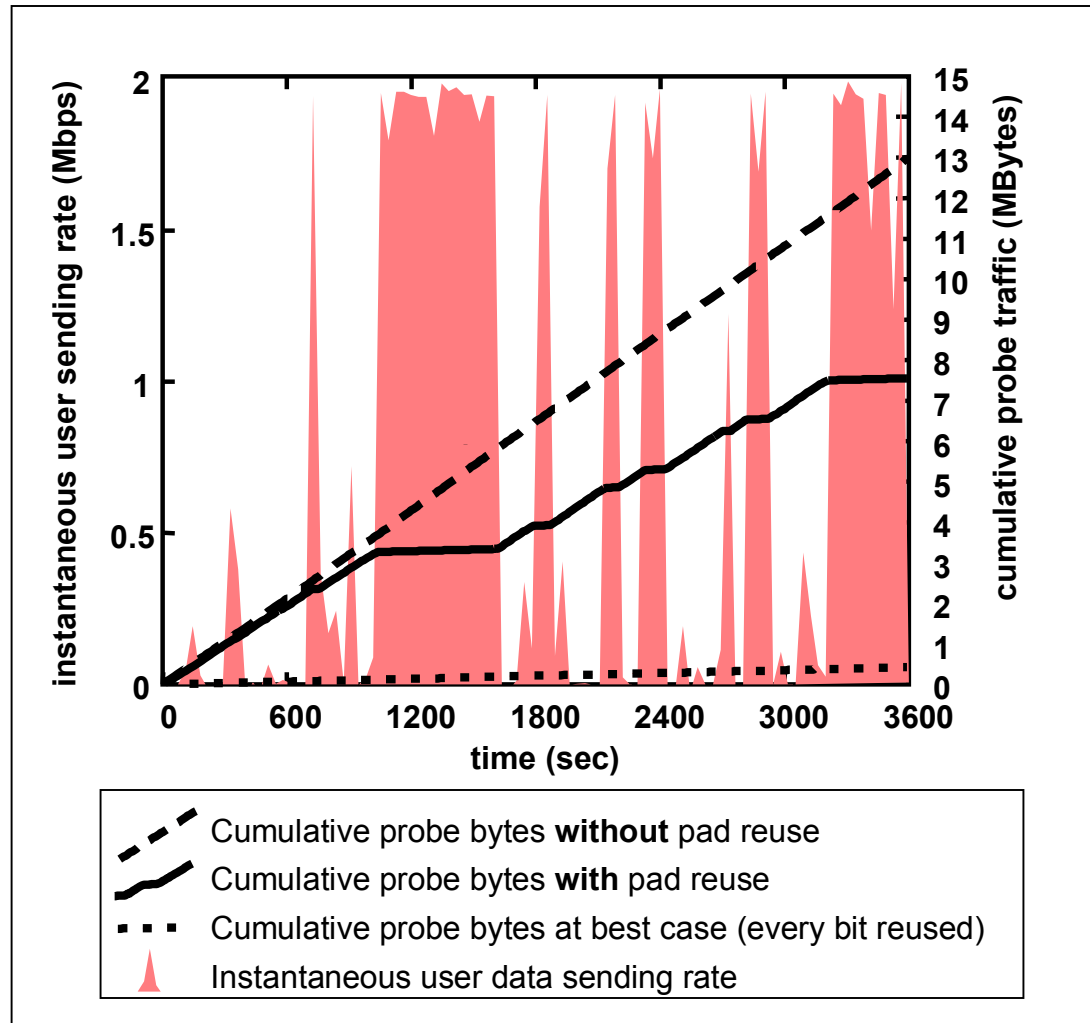
    /* only the first packet of train can be delayed 100msec */
    ctrl_pkt->pkt_delay = (i == 0) ? 100 : 0;

    res = sendmsg(sock, &msg, 0);
    seqno++;
}
```

Preliminary Results

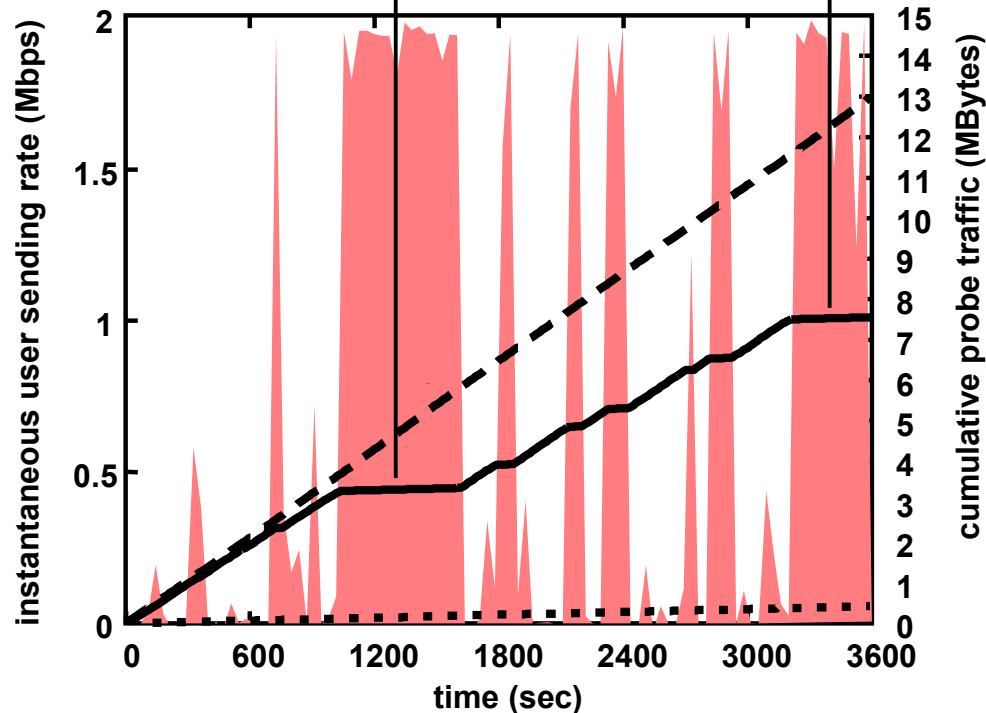
- Proof of concept
- Real traffic on isolated topology – Emulab
- Host **S** replays web file transfers to host **C**
 - 1800 TCP file transfers, 300MB, 1 hour
- Host **S** actively probes the path **SC**
 - 1000-byte packet pair/1sec
 - Packet train of ten 1000-byte packets/10sec
 - Single 500-byte packet/1sec
 - Single 1400-byte packet/2sec

Bandwidth Savings – Heavy Load



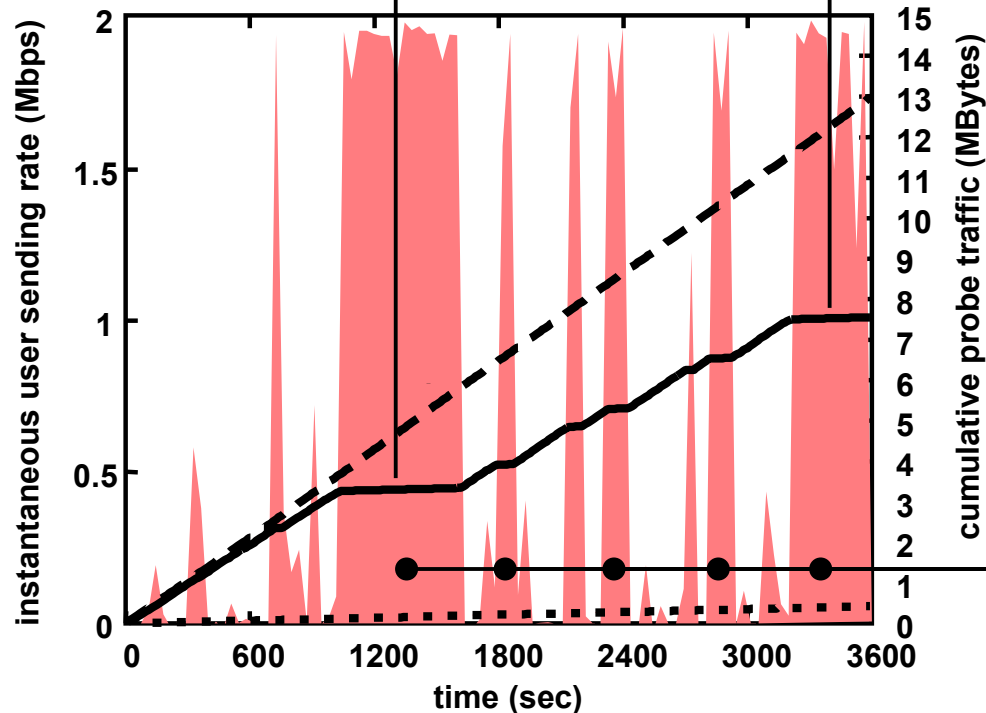
Bandwidth Savings – Heavy Load

Active probing is practically free during heavy load



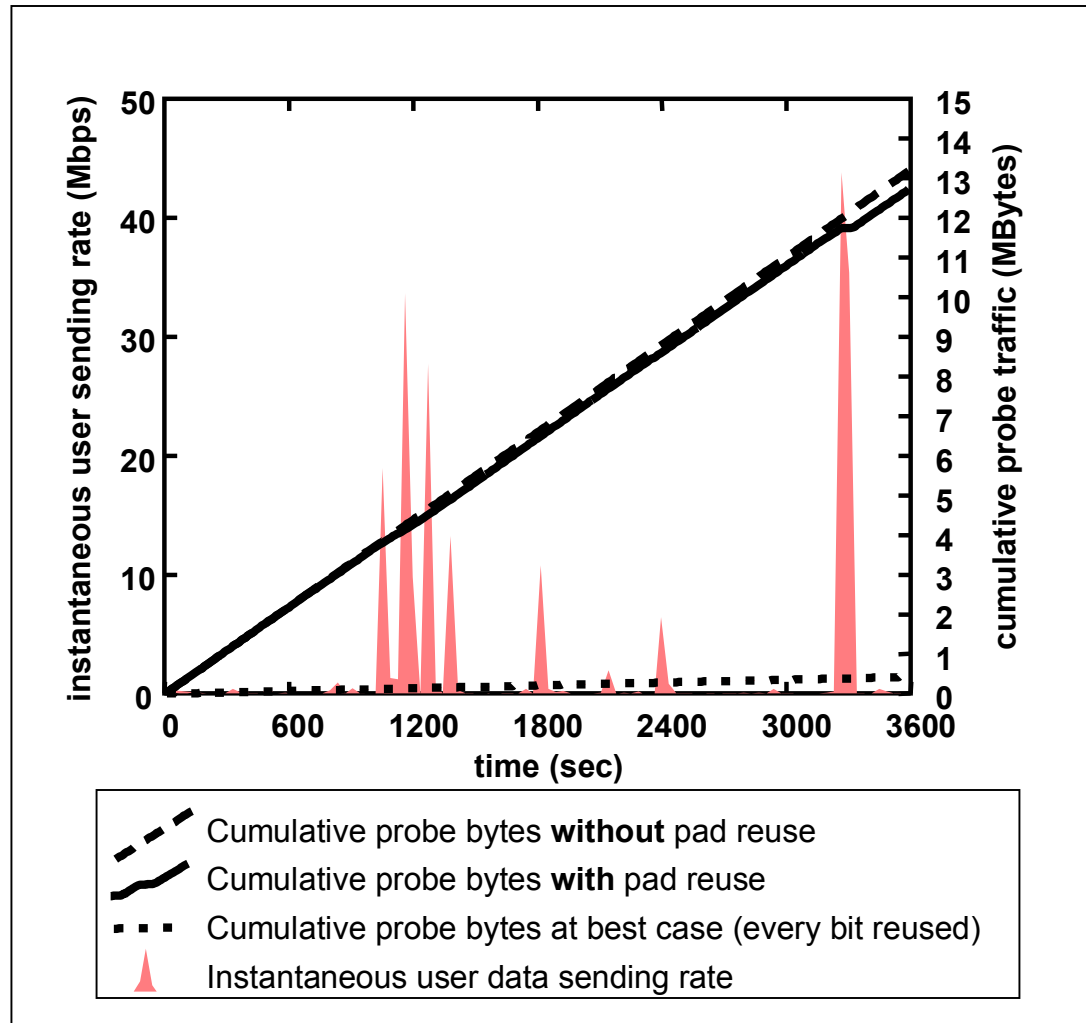
Bandwidth Savings – Heavy Load

Active probing is practically free during heavy load

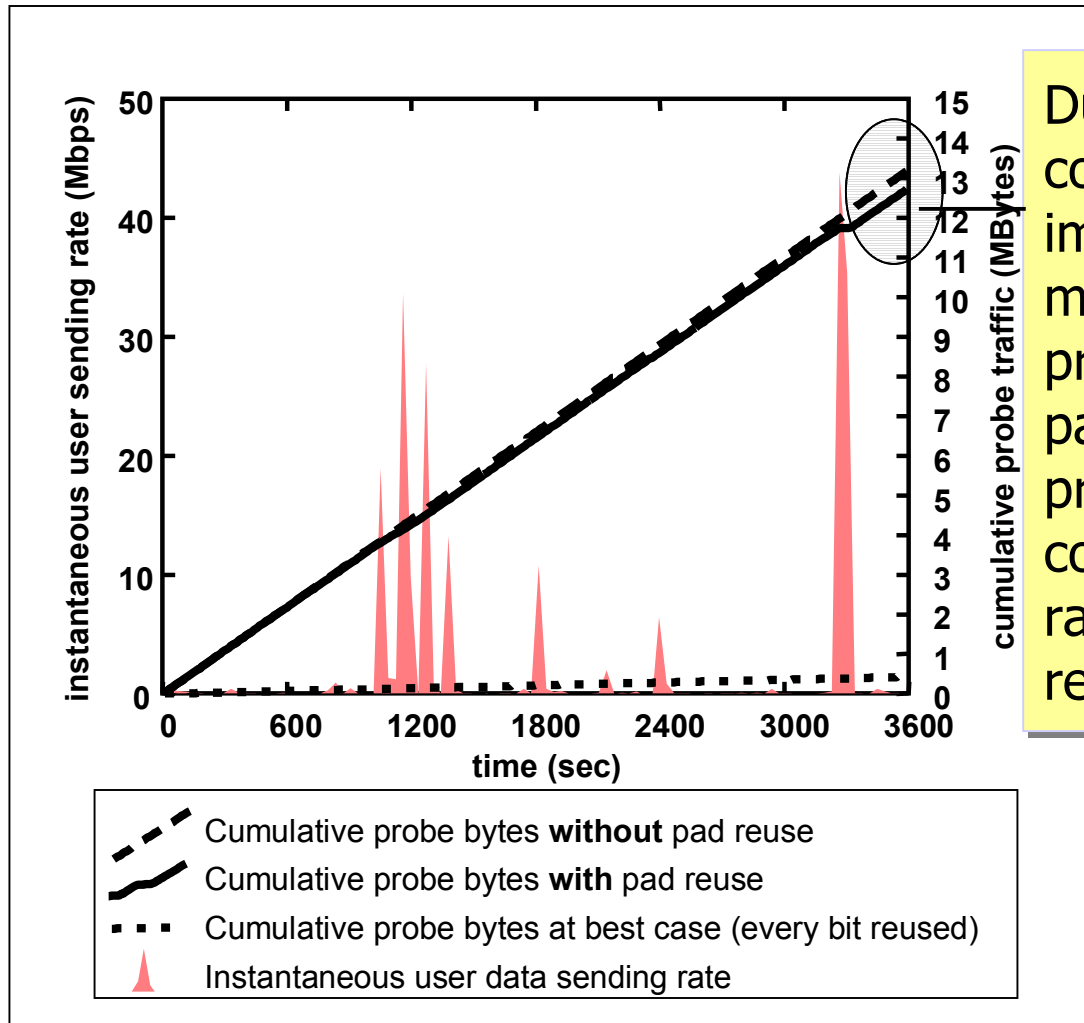


During periods of heavy load there is always user data to send. Probes can reuse ~100% of the padding bits to carry user bits.

Bandwidth Savings – Light Load

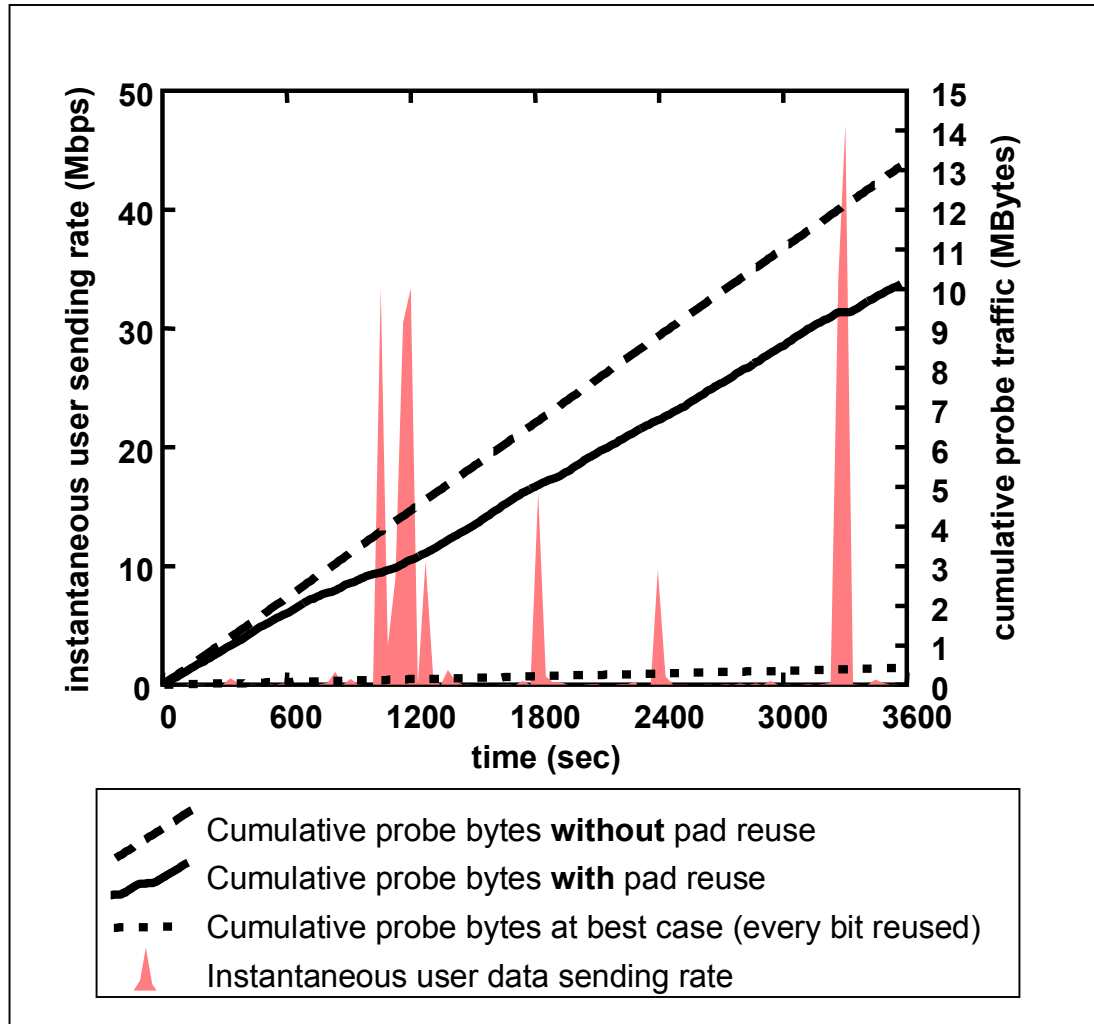


Bandwidth Savings – Light Load

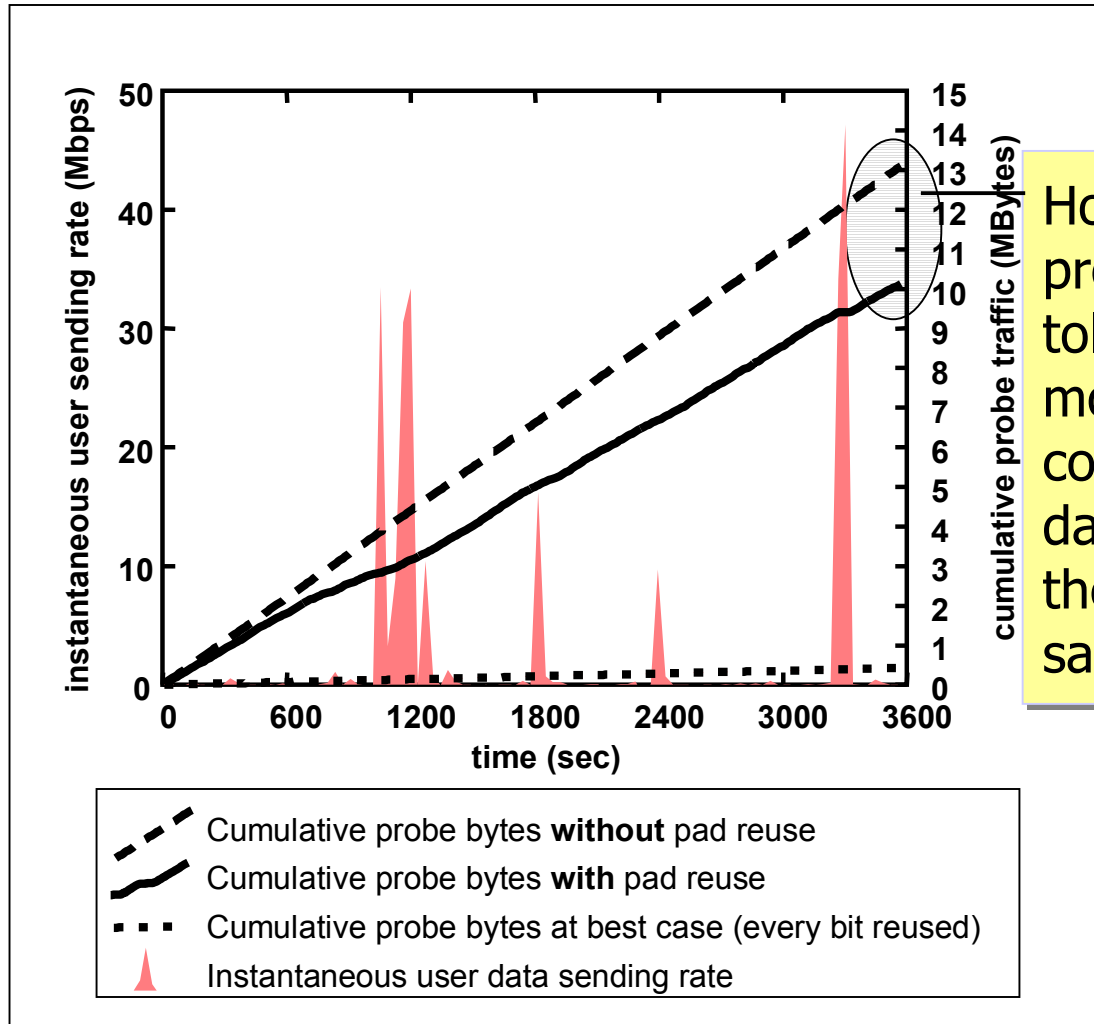


During light load conditions the improvement is marginal because probe and data packets are less probable to coincide and the ratio of pad reuse remains low.

Bandwidth Savings – Delaying Probes



Bandwidth Savings – Delaying Probes



However, if probes can tolerate a delay, more probes coincide with data and hence the bandwidth savings.

Advantages

- Flexibility of **active** probing
- Benefits of **passive** measurement
- Nearly eliminates overhead during conditions of high load
- Same socket API with enhancements
- Expands solution space for tools
 - Bandwidth intensive tools can be considered
 - Hybrid schemes with aggressive probing when user data available, less aggressive when not

Open Issues

- Only end-to-end
- Cannot be used for tools that use ICMP
- Savings only when user packets present
 - Performs **no worse** than traditional case
- Increases the complexity of the transport
- Tampers with packet dynamics and may affect TCP retransmissions
 - Alleviated if TCP data gets encapsulated into a probe only when congestion window is open

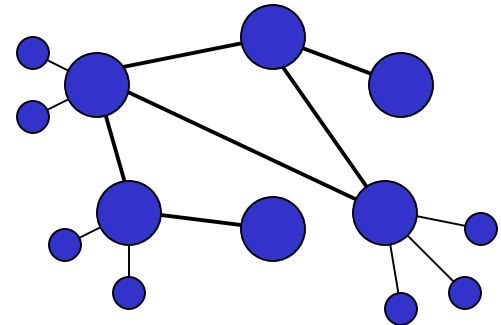
Future Directions

- Run existing tools over modified transport
 - Pathload, Pathchirp, Iperf (avail bandwidth)
 - Shared congestion detection
- Explore new measurement techniques
 - Bandwidth intensive, hybrid techniques
- Measurement primitives in transport
 - **Echo operation** for one-way active probing
 - Send a packet, collect timestamps, report
 - Empower applications, **now we can probe too!**

MediaNet

- Media Streaming Overlay

- Schedules flows using available resources
- Intelligent dropping of packets for controlled quality degradation in face of adverse network conditions



Media Streaming Overlay

<http://www.cs.umd.edu/projects/medianet>

- Network Monitoring

- Uses TCP state to infer state of overlay paths
- Does not monitor paths where no traffic is flowing
- Integrate probe-aware transport to monitor more closely active paths with minimal overhead

Conclusions

- Active measurement too intrusive
- Passive measurement too inflexible
- Gray area in between not exploited
- What if the transport becomes aware of network measurement?
- Can it serve as a knob to go between the two extremes?
- Can it provide a common set of primitives used by all measurement tools?