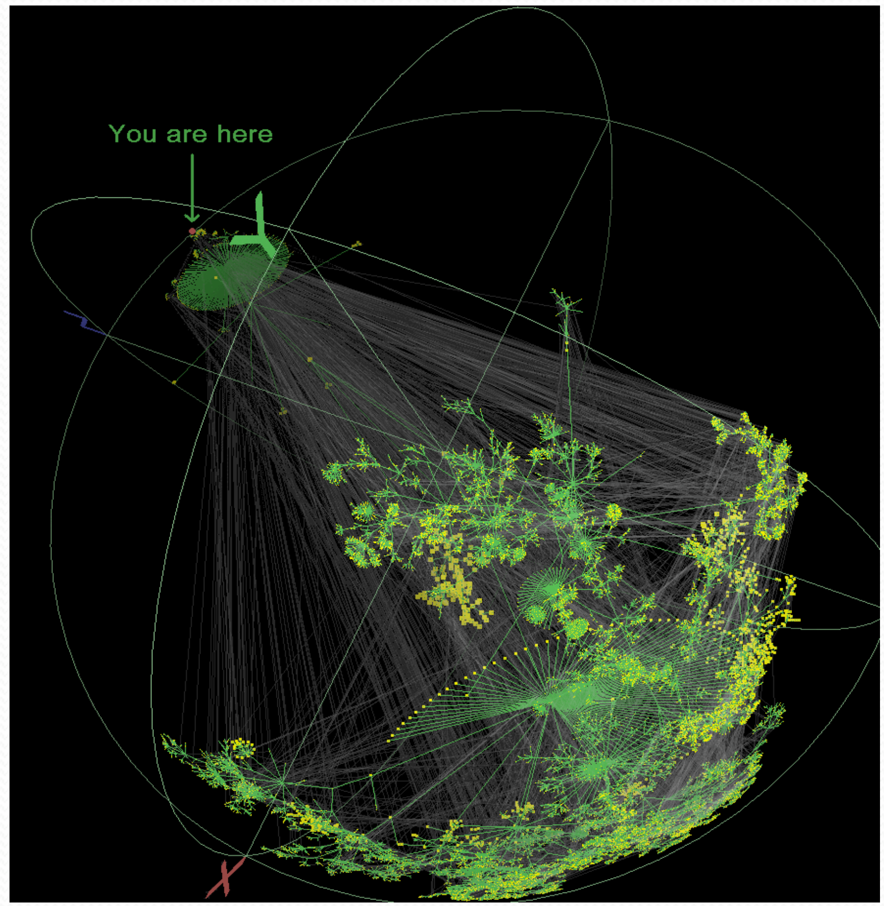
Design and implementation of TCP data probes for reliable and metric-rich network path monitoring

Xiapu Luo, Edmond W. W. Chan, Rocky K. C. Chang
Department of Computing
The Hong Kong Polytechnic University
2009-06-17

Motivations

- How to measure millions of arbitrary paths?
  - Active and non-cooperative
- How to avoid biased measurement samples?
  - TCP data vs. TCP control and ICMP
- How to decrease the measurement overhead?
- How to measure multiple metrics?
- Our answer: OneProbe



The figure is from CAIDA's gallery www.caida.org/tools/visualization/walrus/gallery1/

Content

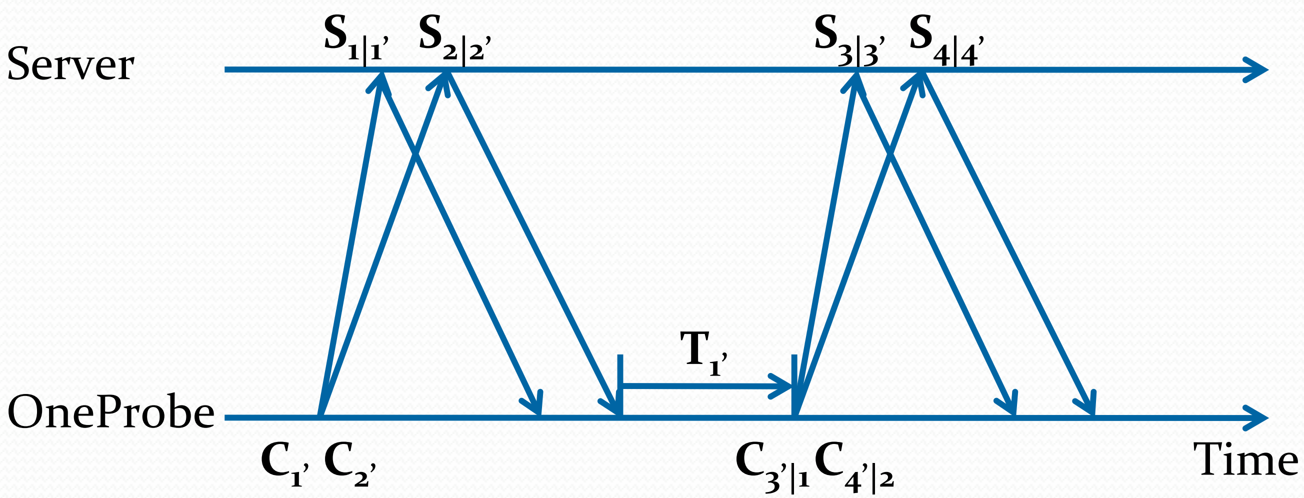
- OneProbe Design
- HTTP/OneProbe
- Evaluation
- Internet path measurement
- Related work
- Conclusions

Design principles

- Measuring data-path quality
  - TCP data packet vs. TCP control packet
    - Firewall
    - Size
- Using multiple metrics
  - Loss, RTT, Packet reordering
- Separating forward/reverse-path measurement
  - Forward path: Measuring node to remote server
- Extensible
  - Different sampling processes
  - New metrics
- Compatibility
  - OneProbe exploits only basic mechanisms in TCP.
    - Sequence number (SN), Acknowledgement number (AN), Advertising window, Maximum segment size (MSS), Flags.

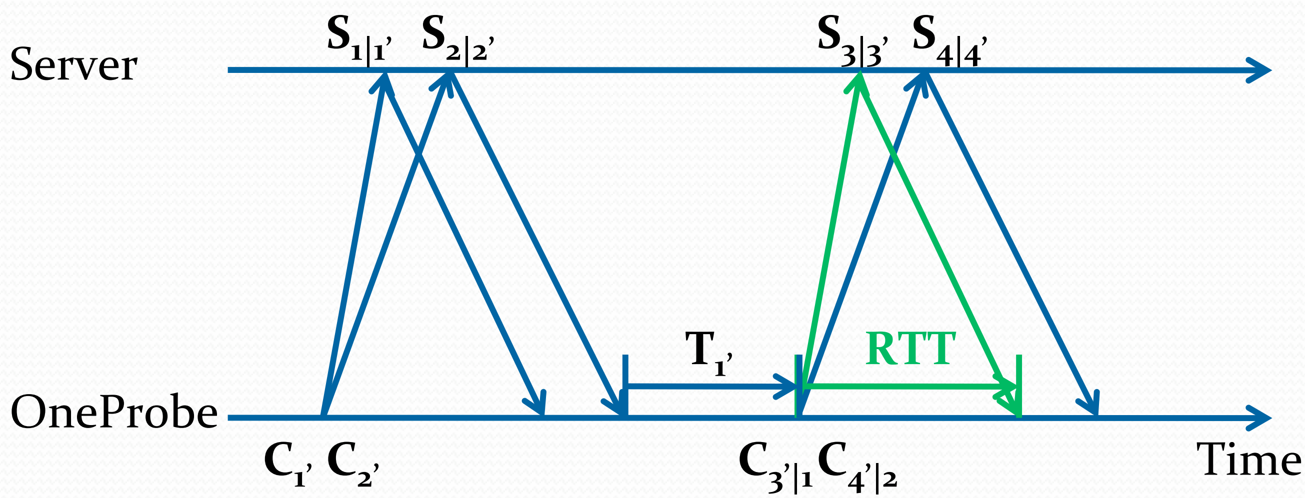
Probing process

- Notations
  - $C_{m|n}$: a probe packet with SN=m and AN=n
  - $S_{m|n}$: a response packet with SN=m and AN=n
- An example



Measuring RTT

- The time between sending a probe packet and receiving its induced new data packet.
  - $C_{3'|1}$ <-> $S_{3|3'}$



Detecting packet loss and reordering

- Five possible events on the forward path

Cases	First probe packet	Second probe packet	Receive order
F0	✓	✓	Same order
FR	✓	✓	Reordered
F1	✗	✓	N.A.
F2	✓	✗	N.A.
F3	✗	✗	N.A.

- Five similar possible events on the reverse path
  - R0, RR, R1, R2, and R3

Identify different events (I)

- The 18 possible loss-reordering events
  - 17 events indicated ✓ and one event for F3
  - Events denoted by – are not possible.

	R0	RR	R1	R2	R3
F0	✓	✓	✓	✓	✓
FR	✓	✓	✓	✓	✓
F1	✓	✓	✓	✓	✓
F2	✓	–	✓	–	–
F3	–	–	–	–	–

Identify different events (II)

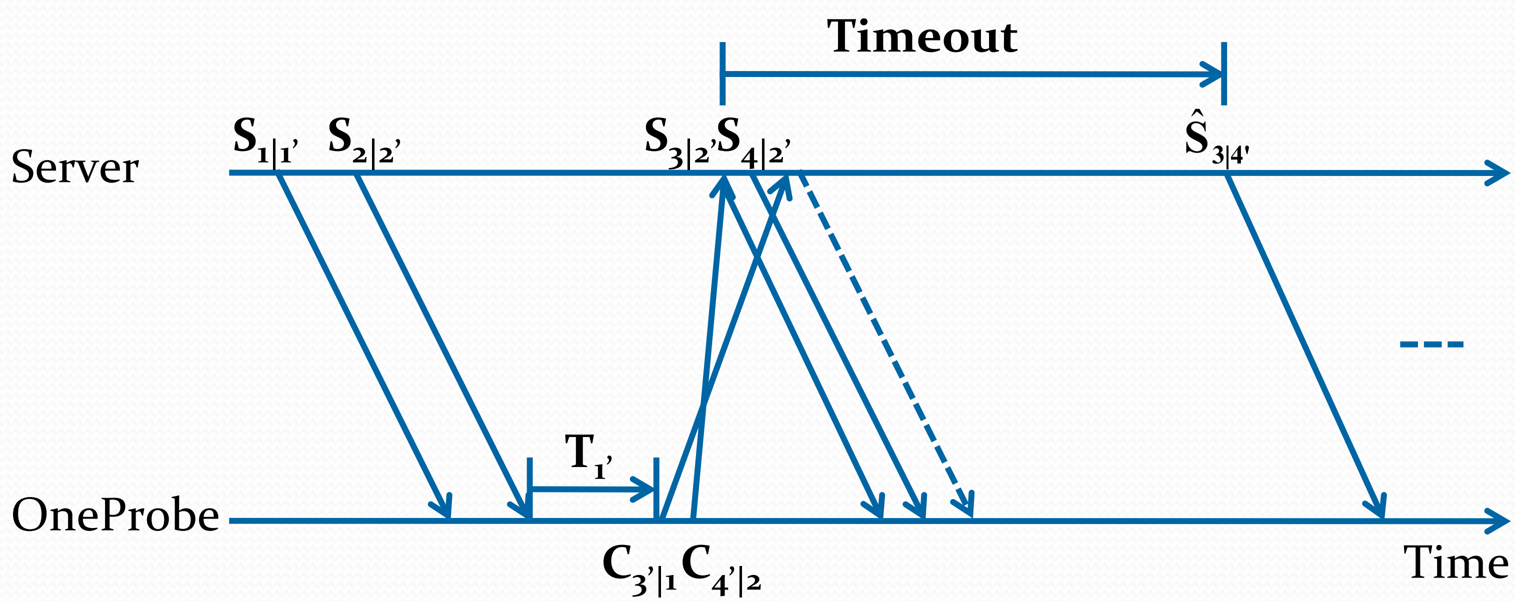
- Information used to distinguish them
  - SN, AN of response packets and retransmitted packets

The response packets induced by the $\{C3'|1, C4'|2\}$ probe for the 18 path events according to RFC 793.

Path events	1st response packets	2nd response packets	3rd response packets
1. F0×R0	$S3\|3'$	$S4\|4'$	–
2. F0×RR	$S4\|4'$	$S3\|3'$	–
3. F0×R1	$S4\|4'$	$\widehat{S}3\|4'$	–
4. F0×R2	$S3\|3'$	$\widehat{S}3\|4'$	–
5. F0×R3	$\widehat{S}3\|4'$	–	–
6. FR×R0	$S3\|2'$	$S4\|2'$	$\widehat{S}3\|4'$
7. FR×RR	$S4\|2'$	$S3\|2'$	$\widehat{S}3\|4'$
8. FR×R1	$S4\|2'$	$\widehat{S}3\|4'$	–
9. FR×R2	$S3\|2'$	$\widehat{S}3\|4'$	–
10. FR×R3	$\widehat{S}3\|4'$	–	–
11. F1×R0	$S3\|2'$	$S4\|2'$	$\widehat{S}3\|2'$
12. F1×RR	$S4\|2'$	$S3\|2'$	$\widehat{S}3\|2'$
13. F1×R1	$S4\|2'$	$\widehat{S}3\|2'$	–
14. F1×R2	$S3\|2'$	$\widehat{S}3\|2'$	–
15. F1×R3	$\widehat{S}3\|2'$	–	–
16. F2×R0	$S3\|3'$	$\widehat{S}2\|3'$	–
17. F2×R1	$\widehat{S}2\|3'$	–	–
18. F3	$\widehat{S}1\|2'$	–	–

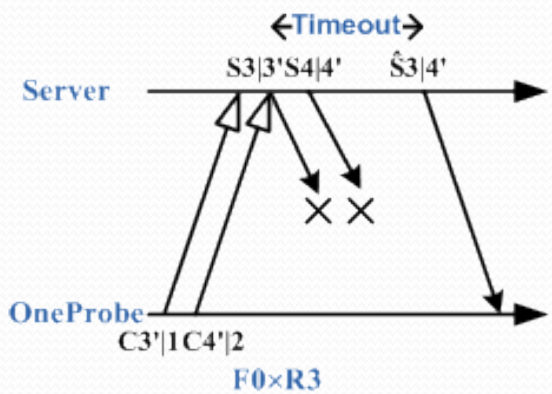
Example

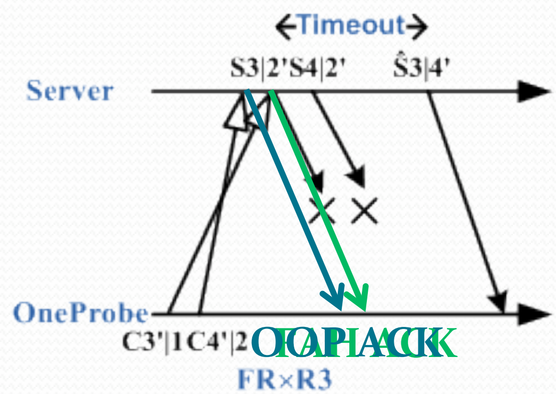
- Forward-path reordering only (FR*Ro)



Distinguish ambiguous events

- F0*R3 vs. FR*R3





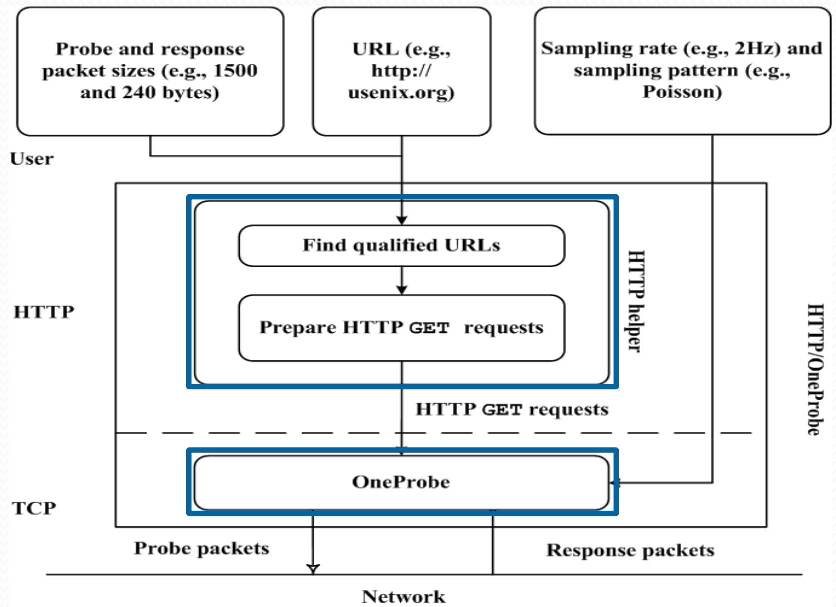
- Solution:
  - Use the filling-a-hole (FAH) ACK triggered by reordered C3'|1.
  - Use the out-of-ordered-packet (OOP) ACK induced by reordered C4'|2 would be used if the server replies it.
  - If the server supports TCP timestamp, $\hat{S}_{3|4'}$'s timestamp will be :
    - Timestamp of C4' in case of F0*R3
    - Timestamp of C3' in case of FR*R3

Content

- OnePobe Design
- HTTP/OneProbe
- Evaluation
- Internet path measurement
- Related work
- Conclusions

Architecture (I)

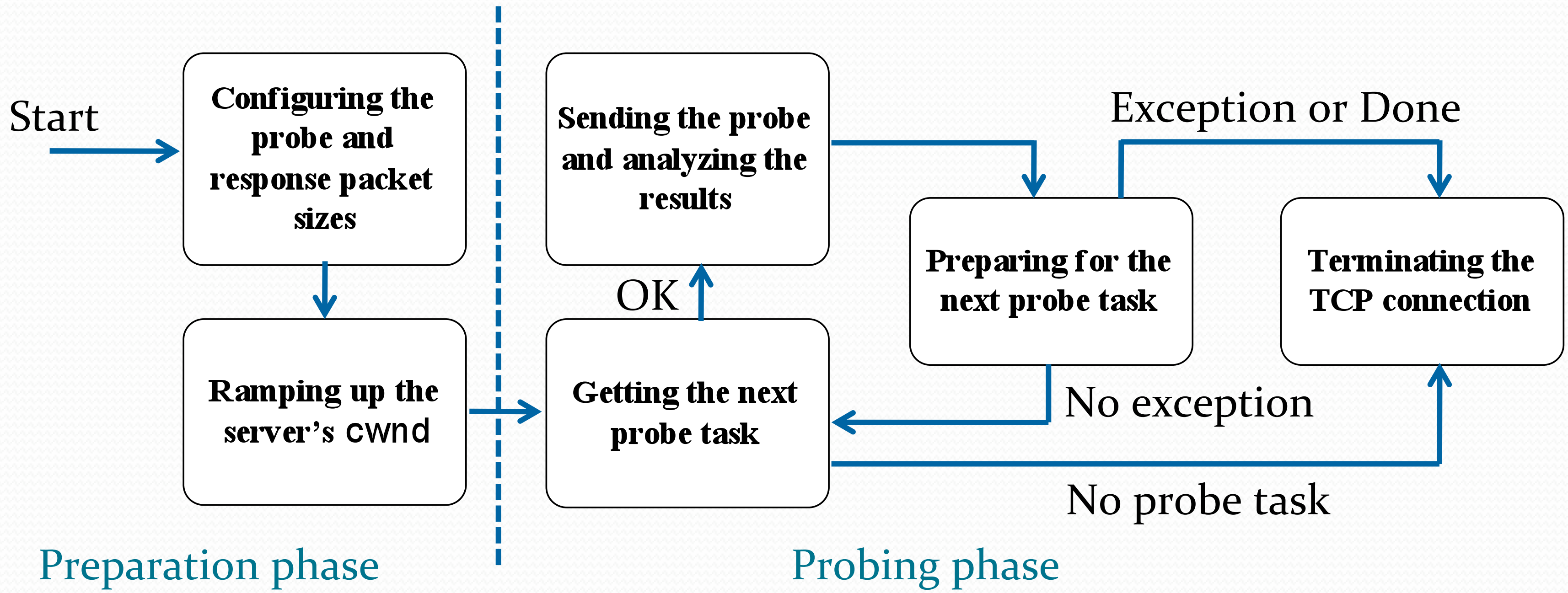
- Implementation
  - User-level tool on Linux 2.6
  - Around 8000 lines of C code
- HTTP helper
  - Find qualified URLs
    - At least five response packets
    - Avoid message compression
      - Accept-Encoding:identity;q=1, *;q=0
    - Range
  - Prepare HTTP GET requests
    - Expand the packet size through the *Referer* field.



Architecture (II)

- OneProbe
  - Manage measurement sessions
    - Connection pool
    - Sampling pattern: periodic, Poisson, etc.
    - Sampling rate
  - Preparation phase and probing phase
    - Negotiate packet size
    - Help a server to increase its congestion window (cwnd)
  - Self-Diagnosis
    - Have the probing packets been sent?
    - Are the response packets dropped due to insufficient buffer space?

Procedure


Start
Configuring the probe and response packet sizes
Ramping up the server's cwnd
Sending the probe and analyzing the results
OK
Getting the next probe task
Exception or Done
Preparing for the next probe task
Terminating the TCP connection
No exception
No probe task
Preparation phase
Probing phase

Content

- OneProbe Design
- HTTP/OneProbe
- Evaluation
- Internet path measurement
- Related work
- Conclusions

Validation

- Four validation tests
  - V0, VR, V1, V2 <-> F0, FR, F1, F2
- 39 operation systems and 35 Web server software
- Test 37,874 websites
  - Successful 93%
  - Fail in the preparation phase 1.03%
  - Fail in V0 0.26%
  - Fail in VR 5.71%

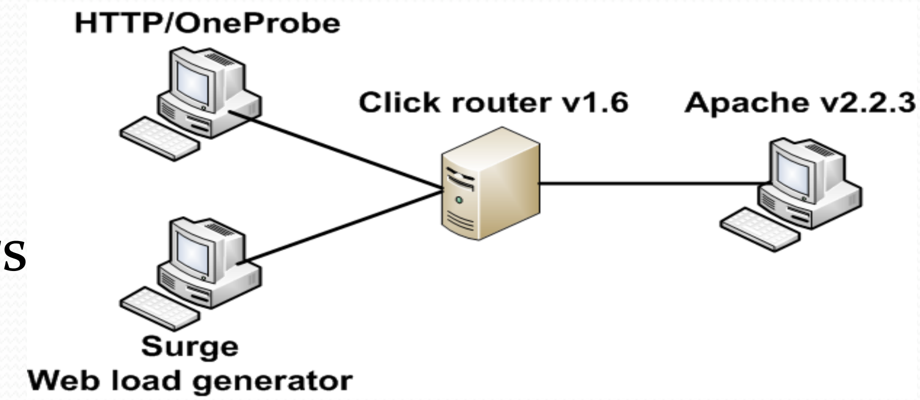
The 39 systems and 35 web server software that passed the OneProbe validation tests.

Systems tested in our lab.:	FreeBSD v4.5/4.11/5.5/6.0/6.2, Linux kernel v2.4.20/2.6.5/2.6.11/2.6.15/2.6.18/2.6.20, MacOSX 10.4 server, NetBSD 3.1, OpenBSD 4.1, Solaris 10.1, Windows 2000/XP/Vista
Systems tested in the Internet:	AIX, AS/400, BSD/OS, Compaq Tru64, F5 Big-IP, HP-UX, IRIX, MacOS, NetApp NetCache, NetWare, OpenVMS, OS/2, SCO Unix, Solaris 8/9, SunOS 4, VM, Microsoft Windows NT4/98/Server 2003/2008
Servers tested in our lab.:	Abyss, Apache, Lighttpd, Microsoft IIS, Nginx
Servers tested in the Internet:	AOLserver, Araneida, Apache Tomcat, GFE, GWS-GRFE, IBM HTTP Server, Jetty, Jigsaw, LiteSpeed, Lotus-Domino, Mongrel, Netscape-Enterprise, OmniSecure, Oracle HTTP Server, Orion, Red Hat Secure, Redfoot, Roxen, Slinger, Stronghold, Sun Java System, thttpd, Twisted Web, Virtuoso, WebLogic, WebSiphon, Yaws, Zeus, Zope

We use Netcraft's database to identify operating systems and Web servers found in the Internet .

Test bed experiments

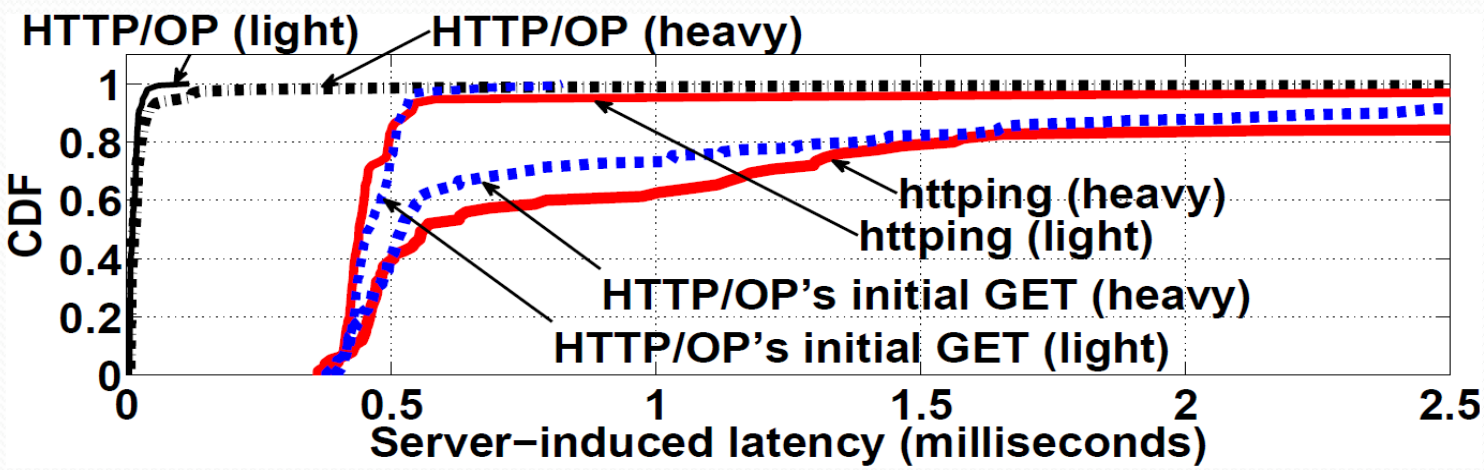
- Setup
  - Light load: 20 Surge users
  - High load: 260 Surge users



- Major observations
  - By avoiding the start-up latency, the HTTP/OneProbe's RTT measurement is much less susceptible to server load and object size.
  - HTTP/OneProbe's CPU and memory consumption in both the probe sender and web server is very low.

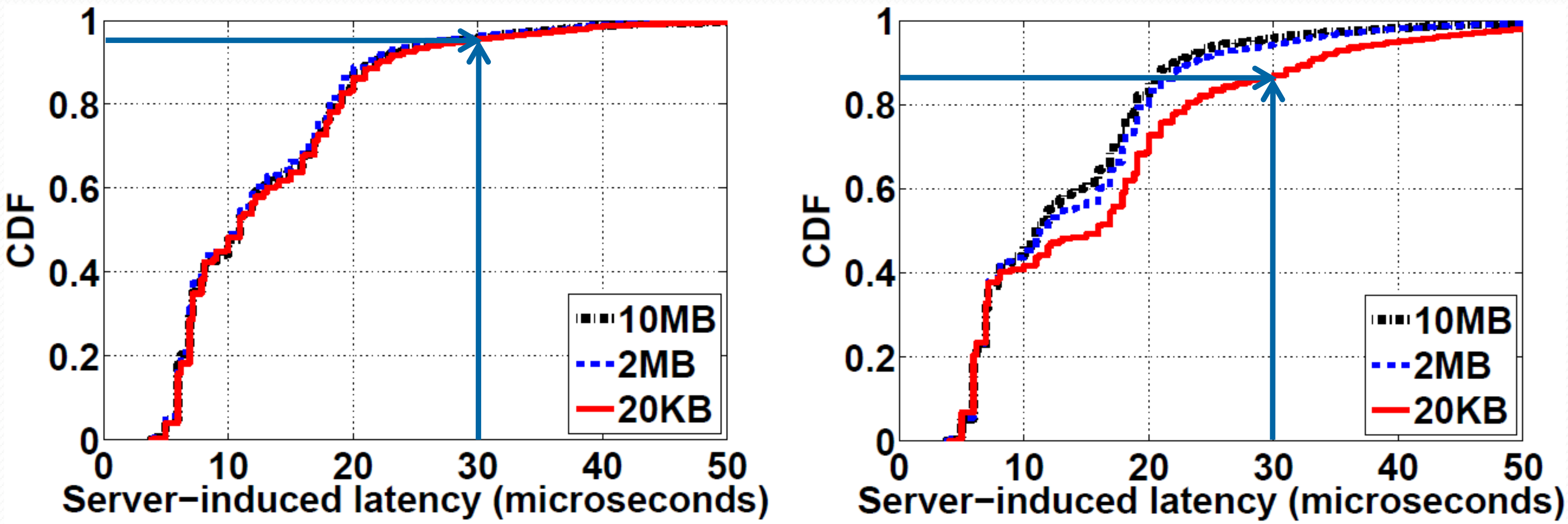
Server induced latency

- HTTP/OneProbe
  - 30 TCP connections and sampling rate 20Hz
  - Size of probe and response packets: 240 bytes
- HTTPing
  - HEAD request
  - Default sampling rate 1Hz
  - Packet size depends on URL and the corresponding response.
- Metric
  - Period between receiving a probe and sending out the first response packet



Effect of object size

- Server induced latency



(a) Light server load

(b) Heavy server load

System resources consumptions

- Fetch a 61M object for 240 seconds with different number of TCP connections and sampling rates.
- Size of probe and response packets is 1500 byte.

The CPU utilizations consumed in the probe sender and web server during the HTTP/OP measurement.

Number of TCP connections	Sampling rates (Hz)	Average CPU utilizations (%)	
		Probe sender	Web server
1	1	<0.01	0.03
1	5	0.07	0.07
10	10	<0.01	0.27
10	50	0.07	0.70
100	100	0.17	0.77
100	150	0.87	1.17

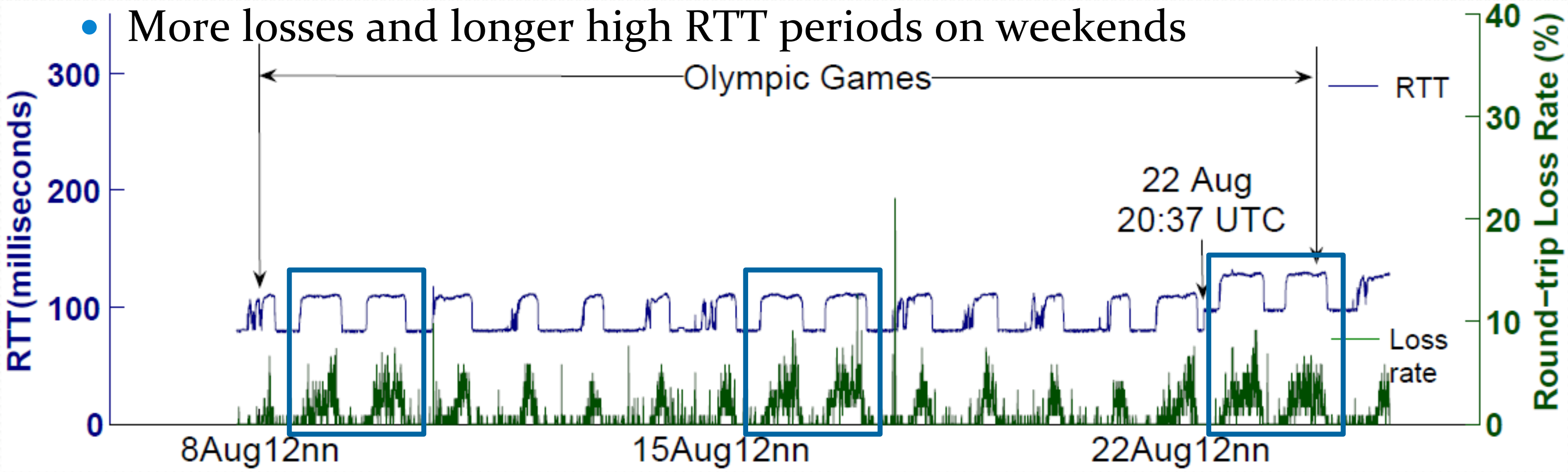
- Average memory utilizations of the probe sender and web server were less than 2% and 6.3% in all cases.

Content

- OneProbe Design
- HTTP/OneProbe
- Evaluation
- Internet path measurement
- Related work
- Conclusions

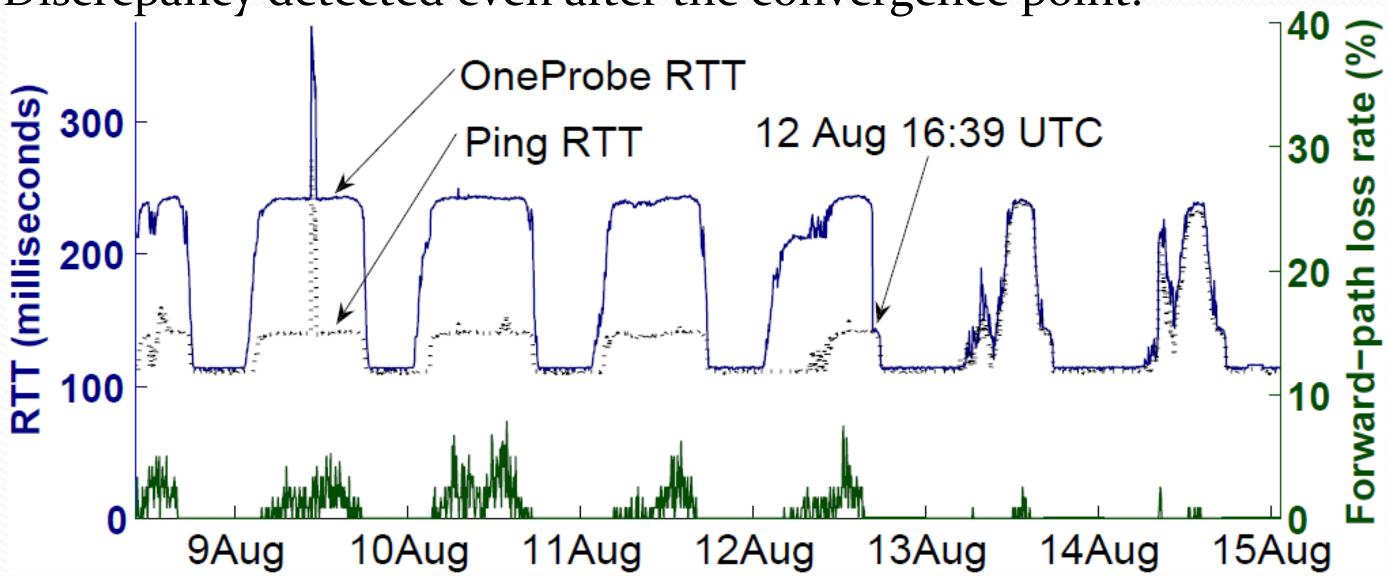
Diurnal RTT and Loss patterns

- Web servers hosting the Olympic Games'08
  - Conduct periodic sampling (2HZ) for one minute and then become idle for four minutes in order to be less intrusive
  - Path: HK (5)->AP-TELEGLOBE (2)->CNCGroup Backbone (4) -> Beijing Province Network (4)
- Observations
  - Diurnal RTT and round-trip loss patterns
  - Positive correlation between RTT and loss rate
  - More losses and longer high RTT periods on weekends



Discrepancy between Ping and OneProbe RTTs

- Path: HK (5)->Korea(2)->CNCGroup Backbone(4)->Henan Province Network(5)
- Observations:
  - RTT consistently differed by around 100 ms during the peaks for the first 4 days.
  - They were similar in the valleys.
  - Their RTTs “converged" at 12 Aug. 2008 16:39 UTC (~1.5 hrs into the midnight).
  - Discrepancy detected even after the convergence point.



Related work

- Sting
  - Seminal work on TCP-based non-cooperative measurement
  - Measure loss rate on both forward path and reverse path
  - Unreliable due to anomalous probe traffic (a burst of out-of-ordered TCP probes with zero advertised window)
  - Lack of support for variable response packet size
- Tulip
  - Hop-by-hop measurement tool based on ICMP
  - Locate packet loss and packet reordering events and measure queuing delay.
  - Require routers or hosts support consecutive IPID.
- TCP sidecar
  - Inject measurement probes in a non-measurement TCP connection.
  - Cannot measure all loss scenarios
  - Cannot control sampling pattern and rate.
- POINTER
  - Measure packet reordering on both forward path and reverse path
  - Unreliable due to anomalous probe traffic (unexpected SN and AN)

Conclusions

- Proposed a new TCP-based non-cooperative method
  - Reliable
  - Metric rich
- Implemented HTTP/OneProbe and conduct extensive experiments in both test bed and Internet.
  - **www.oneprobe.org**
- Future work
  - Add new path metrics, e.g. capacity, available bandwidth, etc.
  - Server-side OneProbe for opportunistic measurement.
  - Implement OneProbe into other TCP-based applications, e.g. P2P, video, etc.

Acknowledgement

- This work is partially supported by a grant (ref. no. ITS/152/08) from the Innovation Technology Fund in Hong Kong.

THANKS