

## An Approach for Non-Intrusive Round Trip Time Measurements

Brandon Tyo, Sebastian Zander  
{btyo,szander}@swin.edu.au



### Overview

- Motivation
- Wireless Gaming Testbed
- Delay Measurement Approaches
- OpenIMP
- Passive RTT Algorithm
- Evaluation
- Conclusions & Future Work



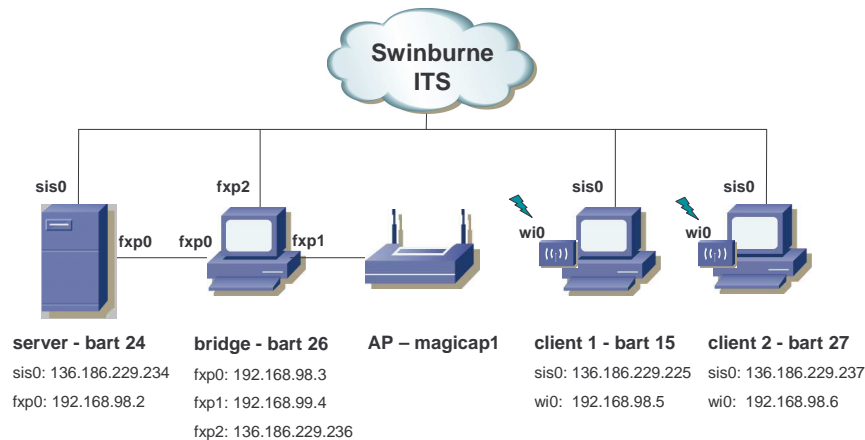


## Motivation

- Monitor game traffic over wireless 802.11b,g access network
- Measure delay, loss, throughput with respect to the number of game clients and collisions in the WLAN



## Wireless Gaming Testbed



## Delay Measurement Approaches



### ■ Passive One Way Delay

- ☑ Do not send additional packets – only monitor packets (non-intrusive)
- ☑ Allows to measure delay in each direction (e.g asymmetric paths)
- ☑ Requires accurate time synchronisation. NTP synchronisation is too inaccurate for measuring small testbed delays, GPS is too expensive.

### ■ Active Round Trip Time (e.g. ping)

- ☑ Additional measurement traffic influences network under test
- ☑ Does not measure the actual game traffic
- ☑ Routers may treat measurement packets differently (e.g ICMP)
- ☑ Simple ping needs only control on one end host



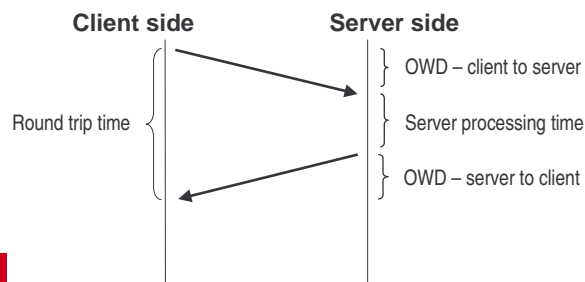
<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 5

## Delay Measurement Approaches



### ■ Passive Round Trip Time

- ☑ Use of packet pairs requires NO time synchronisation
- ☑ Non-intrusive, measures actual game traffic packet
- ☑ Need control on both end hosts/probes (no problem in testbed!)
- ☑ Don't know OWDs
- **Best accuracy/cost**



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 6

## OpenIMP – Overview

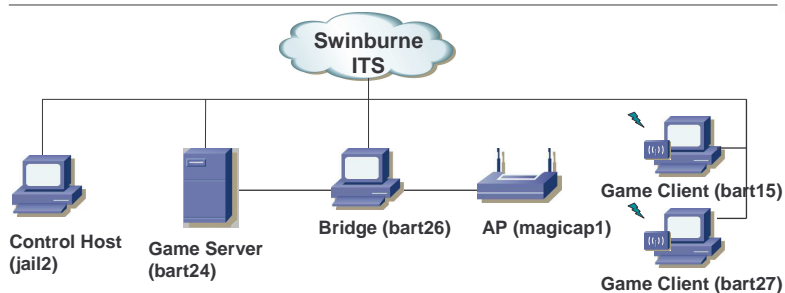


- OpenIMP monitoring software (Unix based)
  - <http://www.ip-measurement.org/openimp/>
- What is it?
  - Passive measurement of IP traffic
  - Supports metrics such as volume, one-way-delay, jitter and packet loss
  - Web-based GUI
    - Setup measurements
    - Display results (delay and volume time series and distributions)



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 7

## OpenIMP – In The Testbed



- One control host (jail2), multiple measurement probes (bart24, bart15 and bart27)
- Measurement tasks are configured on the control host and sent to the probes.
- Probes sent back measurement data to the controller, which then performs delay computation, displays and stores the results
- Not completely non-intrusive (probes running on server/clients)



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 8

## Passive RTT Algorithm



- Python script for post-processing the OpenIMP measurement data
- Identifies packet pairs and computes the RTTs
  - Each packet cannot be not used for more than one pair
  - There is no third packet in-between the two packets that form a pair
  - Packet pair should be close together in time (<100ms)
- Requires two sets of OWD packet measurements (one in each direction: client to server and server to client)

## Passive RTT Algorithm – Input



OpenIMP measurement results - Microsoft Internet Explorer

Address: http://caia.swin.edu.au/~nimp/show\_results.php?taskId=292&file=/home/nimp/retal/results/t-292.cai

### OpenIMP show results

task details

taskid	292
name	bd1a
server	qoscalc
status	done
command	add_task 292 -a calcrowd task=290, task=291 -m duration=60 gmtStart="2005-03-16_00:55:30" -e target=scp://nimp@caia2.22.nmp/mpspool/t-292
function	calcrowd

back to mtask

t-292.cai-crowd-20050316\_005500.dat

```
# pktid | delay/us | timestamp | timestamp | pktlen | tos1 | tos2 | dvar | flowid
2057923932 | 26165.000000 | 1110934505.470017000 | 1110934505.443852000 | 98 | 0 | 0 | 0.000000 | 1
1880611811 | 2329.000000 | 1110934506.508997000 | 1110934506.511326000 | 98 | 0 | 0 | 0.000000 | 0
693308744 | 2268.000000 | 1110934507.527831000 | 1110934507.530099000 | 98 | 0 | 0 | -61.000000 | 0
2121101715 | 2677.000000 | 1110934508.548390000 | 1110934508.551067000 | 98 | 0 | 0 | 409.000000 | 0
1050603458 | 3192.000000 | 1110934509.568161000 | 1110934509.571353000 | 98 | 0 | 0 | 515.000000 | 0
1766681232 | 2415.000000 | 1110934510.670212000 | 1110934510.672627000 | 98 | 0 | 0 | -777.000000 | 0
1693114542 | 2496.000000 | 1110934511.689053000 | 1110934511.691549000 | 98 | 0 | 0 | 81.000000 | 0
1892043024 | 2654.000000 | 1110934512.706869000 | 1110934512.709523000 | 98 | 0 | 0 | 158.000000 | 0
```

## Passive RTT Algorithm – Timestamps

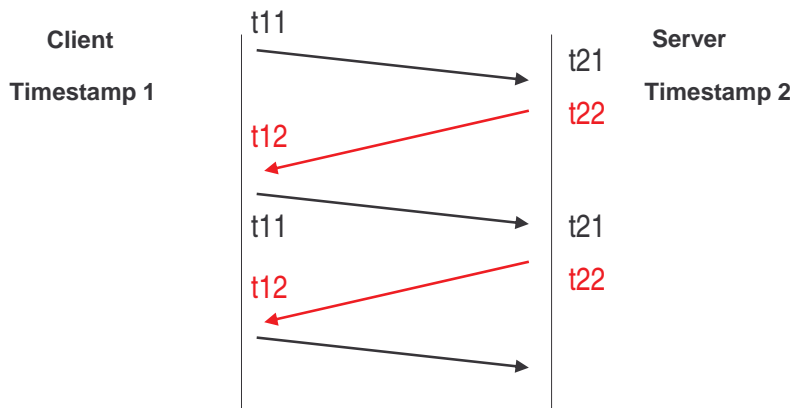


- File information
  - Timestamp 1 = Packet at client
  - Timestamp 2 = Packet at server
- Two files
  - File 1 = client to server OWD
  - File 2 = server to client OWD
- $t$  {location} {direction}
- Four Timestamps
  - $t_{11}$  and  $t_{21}$
  - $t_{12}$  and  $t_{22}$



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 11

## Passive RTT Algorithm – Timestamps



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 12

## Passive RTT Algorithm



- Read file 1 (client to server OWD file) and file 2 (server to client OWD file)
- Pair packets
  - Get next packet in file 1 ( $t_{11}, t_{21}$ )
  - Find next packet in file 2 ( $t_{12}, t_{22}$ ) where  $t_{22} > t_{21}$  (find a pair)
  - Find next packet in file 1 ( $t_{11}^*, t_{21}^*$ ) where  $t_{21}^* > t_{21}$  and  $t_{21}^* < t_{22}$  (find the closest pair)
  - Found pair if  $t_{21} > t_{11}^*$  (no “time traveling” packets)
- Compute
  - $RTT = \overset{\text{Clock 1}}{(t_{12} - t_{11}^*)} - \overset{\text{Clock 2}}{(t_{22} - t_{21}^*)}$
  - Server processing time (SPT) =  $t_{22} - t_{21}^*$
  - Warning if SPT > specified threshold
  - Output:  $\langle t_{11}^*, RTT, SPT \rangle$



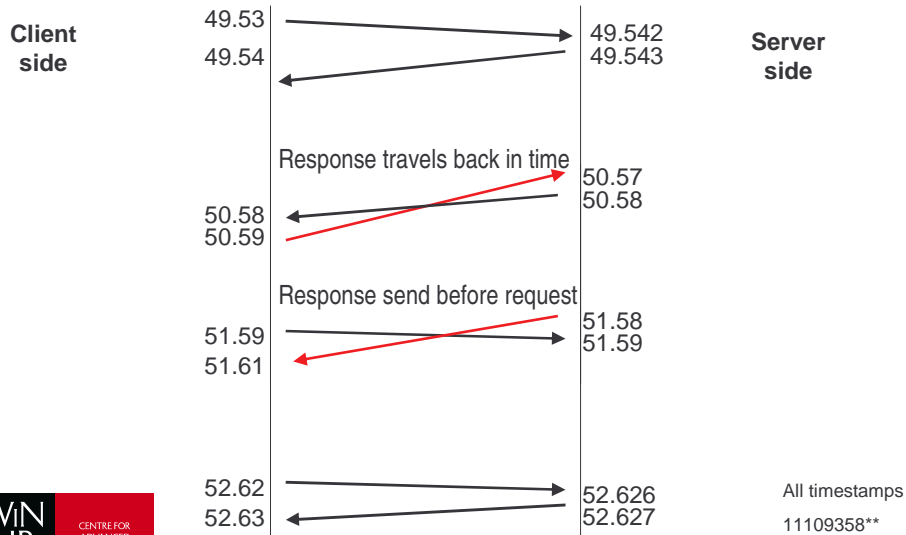
<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 13

## Passive RTT Algorithm



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 14

# Time Stamping Errors



# Passive RTT Algorithm - Output



```

byo@caia.swin.edu.au - PuTTY
byo@bttyo:/home/byo/cprog/python/mtpscriptstuff>./pairOWD.py
Usage: ./pairOWD.py -c file1 -s file2 [-t tolerance] [-r] [-d]
-c file1      : File containing one way delays from client to server
-s file2      : File containing one way delays from server to client
-t tolerance  : Server processing tolerance. Integer in milliseconds
-r           : Reverse. Switches timestamp columns for both files
-d           : Print debug information
byo@bttyo:/home/byo/cprog/python/mtpscriptstuff>./pairOWD.py -s Oitest/bdib.txt
-c Oitest/bdia.txt
#ctl      RTT (ms)      warn
1110934505.470017000 4.05097007751
1110934506.508997000 4.46891784668
1110934507.527831000 3.81779670715
1110934508.548390000 4.38714027405
1110934509.568161000 4.0819644928
1110934510.670212000 3.33189944294
1110934511.689053000 3.5719871521
1110934512.706869000 4.05693054199
1110934513.761760000 3.96800041199
1110934514.785159000 3.33118438721
1110934515.801678000 3.28302383423
1110934516.821142000 3.39102745056
1110934517.884495000 3.74579429626
1110934518.954712000 4.46510314941
1110934519.973554000 3.75604629517
1110934520.991388000 3.73482704163
1110934522.010168000 4.0819644928
1110934523.029053000 3.70719048645
1110934524.047125000 3.73506594624
1110934525.071164000 3.65114212036
1110934526.092981000 3.27587127686
1110934527.116202000 4.12678718567
    
```





## Evaluation

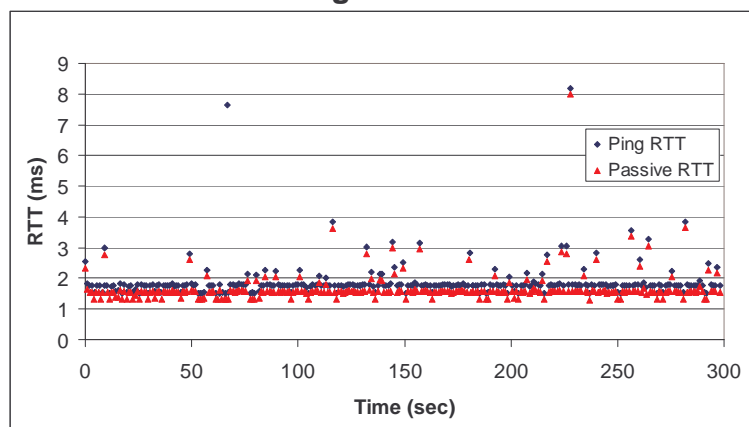


- Experiments using games testbed
  - Measuring traffic between wireless client and server
  - AP and clients in one room, server in adjacent room
- Trial 1: Ping RTT vs. Passive RTT
  - Modified FreeBSD ping to provide send time with micro second precision
  - Python Passive RTT script
- Trial 2: Trial 1 + 100ms dummynet delay (RTT)
- Trial 3: Trial 1 + 200ms dummynet delay (RTT)
- Trial 4: Trial 1 + 400ms dummynet delay (RTT)
- Trial 5: Trial 1 + hand-simulated packet loss
- Game traffic trial: Connecting to ET server and entering game (no movement)

## Evaluation – Trial 1



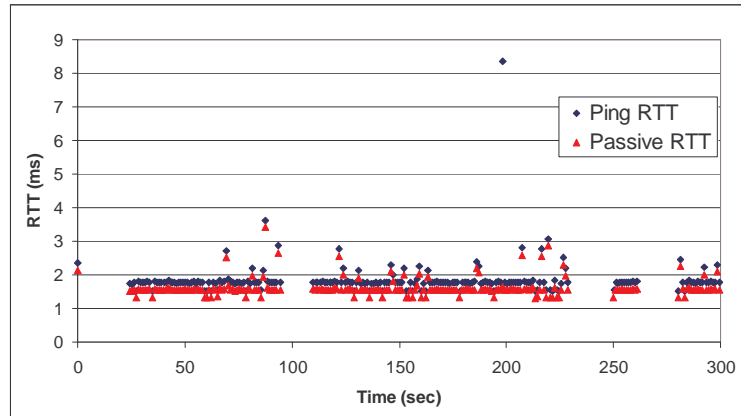
### Passive RTT vs Ping RTT



## Evaluation – Trial 5



### Passive RTT vs Ping RTT

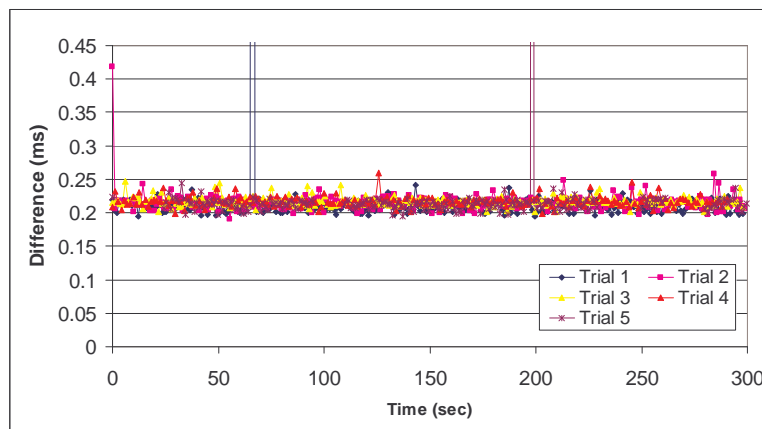


<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 19

## Evaluation – Delay Differences



### Difference between Ping RTT and Passive RTT

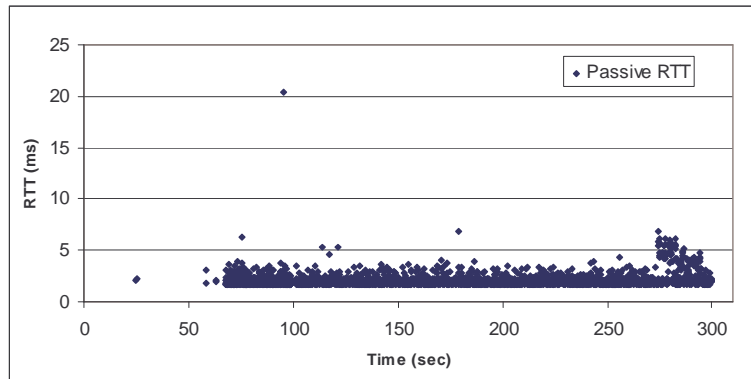


<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 20

# Evaluation – Game Traffic



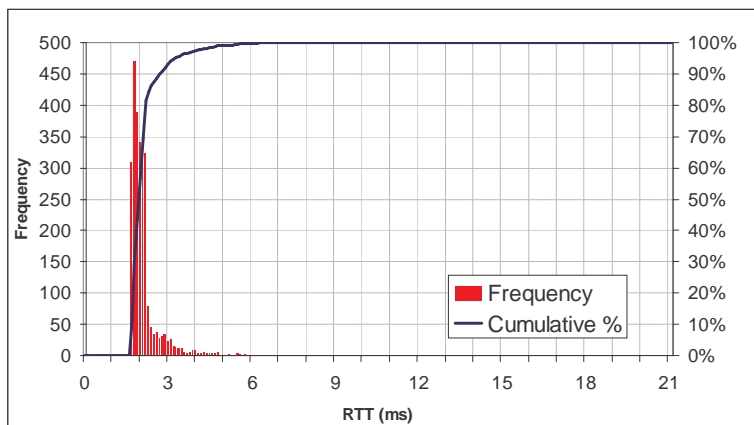
## Passive RTT of ET game traffic over wireless



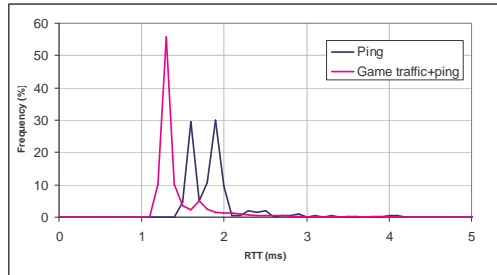
# Evaluation – Game Traffic



## Passive RTT of ET game traffic over wireless

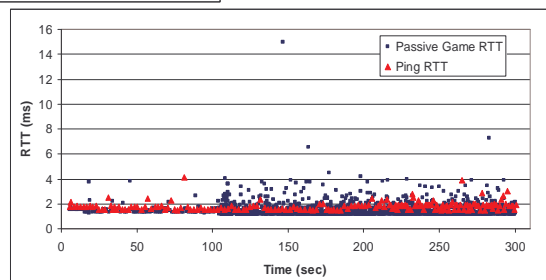


## Evaluation – Game Traffic



Distribution of Ping traffic and Game traffic including Ping traffic

Passive RTT vs Ping RTT



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 23

## Conclusions



- Generally Passive RTT is comparable to Ping RTT
  - Actually Passive RTT is more accurate
  - Ping RTTs include processing time
  - Found very few packet pairs where ping times are significantly larger than passive RTTs
- Pair identification works for game traffic
  - 99% of available (client to server) packets used (counters functionality in script)
  - Server sends a packet approximately every 50ms (Quake engine)
  - Packet rates – server to client (20pps), client to server (fluctuates)



<http://caia.swin.edu.au> btyo@swin.edu.au May 9, 2005 Page 24

## Future Work



### ■ Future work

- Further testing
  - Testing the reliability of the Passive RTT measurement (tcpdump)
  - Higher ET client frame rate
  - Multiple clients
- Game Traffic Measurements
  - 802.11g measurements
  - Performance vs. number of clients
- Processing time measurements
- Enhancing script (e.g implementation of algorithm, etc)
- Promiscuous mode over wireless
- Hardware tester?