SWINBURNE

CENTRE FOR ADVANCED INTERNET ARCHITECTURES

An Approach for Non-Intrusive Round Trip Time Measurements

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

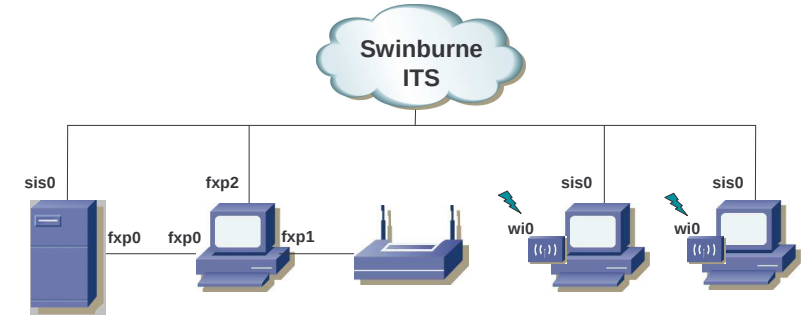
Overview

- 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



server - bart 24
sis0: 136.186.229.234
fxp0: 192.168.98.2

bridge - bart 26
fxp0: 192.168.98.3
fxp1: 192.168.99.4
fxp2: 136.186.229.236

AP – magicap1

client 1 - bart 15
sis0: 136.186.229.225
wi0: 192.168.98.5

client 2 - bart 27
sis0: 136.186.229.237
wi0: 192.168.98.6

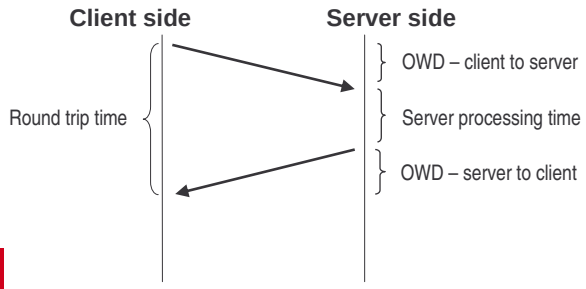
Delay Measurement Approaches

- 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

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

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**

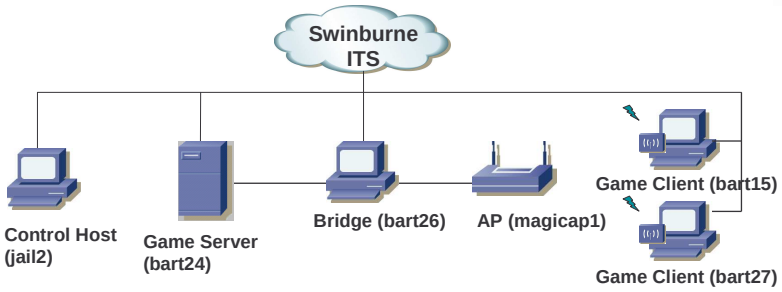


SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

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)

OpenIMP – In The Testbed

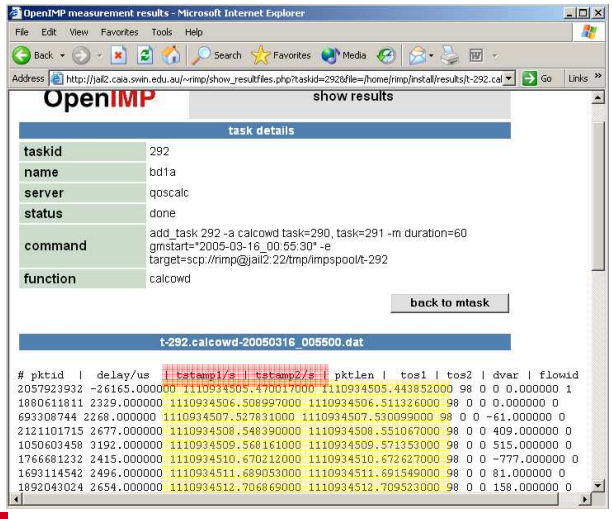


- One control host (jail2), multiple measurement probes (bart24, bart15 and bart27)
- Measurement tasks are configured on the control host and sent to the probes.
- Probes sent back measurement data to the controller, which then performs delay computation, displays and stores the results
- Not completely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

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

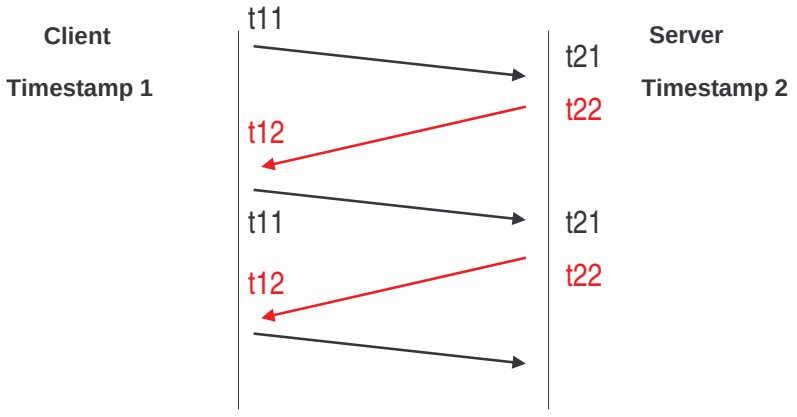


Passive RTT Algorithm – Timestamps

- 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
  - t11 and t21
  - t12 and t22

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Passive RTT Algorithm – Timestamps



SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Passive RTT Algorithm

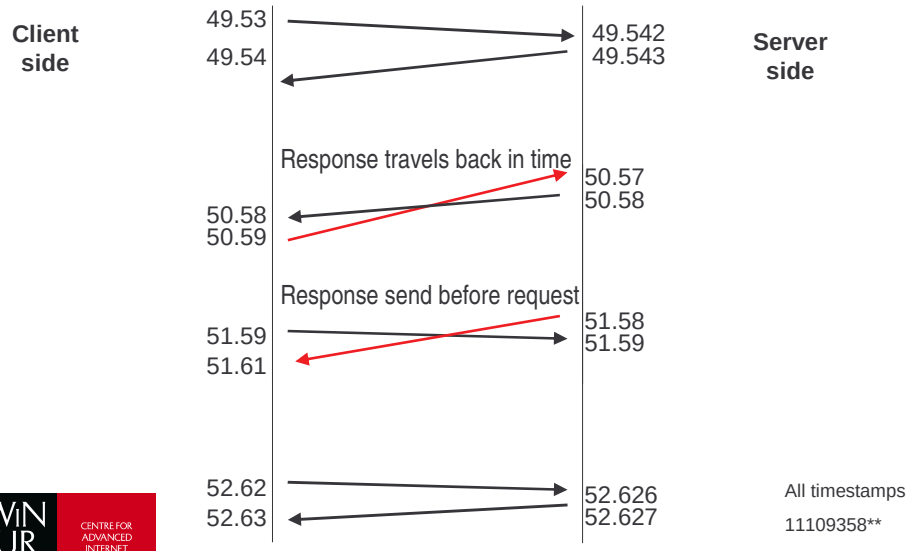
- Read file 1 (client to server OWD file) and file 2 (server to client OWD file)
- Pair packets
  - Get next packet in file 1 (t11,t21)
  - Find next packet in file 2 (t12,t22) where t22 > t21 (find a pair)
  - Find next packet in file 1 (t11*,t21*) where t21*>t21 and t21*<t22 (find the closest pair)
  - Found pair if t21 > t11* (no "time traveling" packets)
- Compute
  - RTT = (t12 – t11*) – (t22 – t21*) (Clock 1 / Clock 2)
  - Server processing time (SPT) = t22 – t21*
  - Warning if SPT > specified threshold
  - Output: <t11*,RTT,SPT>

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

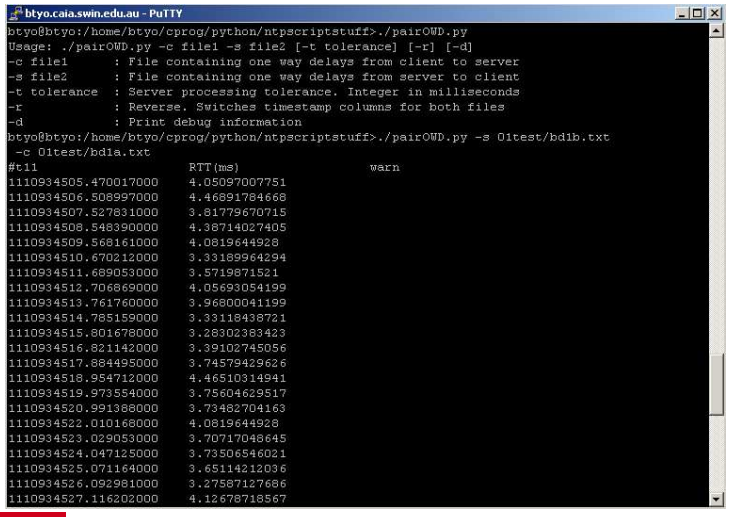
Passive RTT Algorithm

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Time Stamping Errors



Passive RTT Algorithm - Output

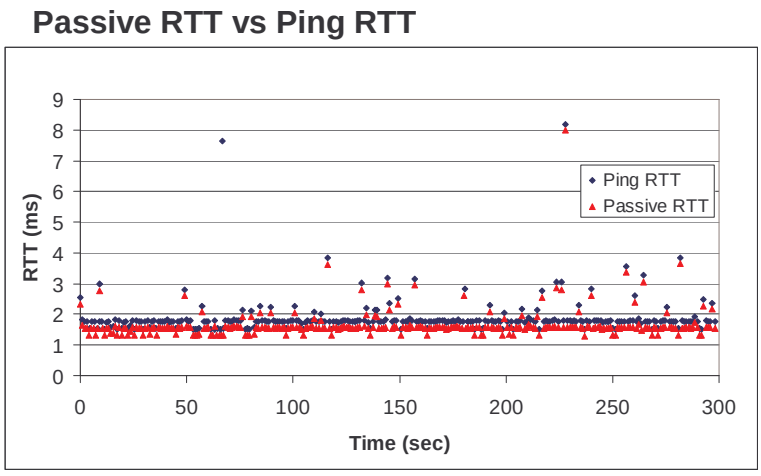


Evaluation

- 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)

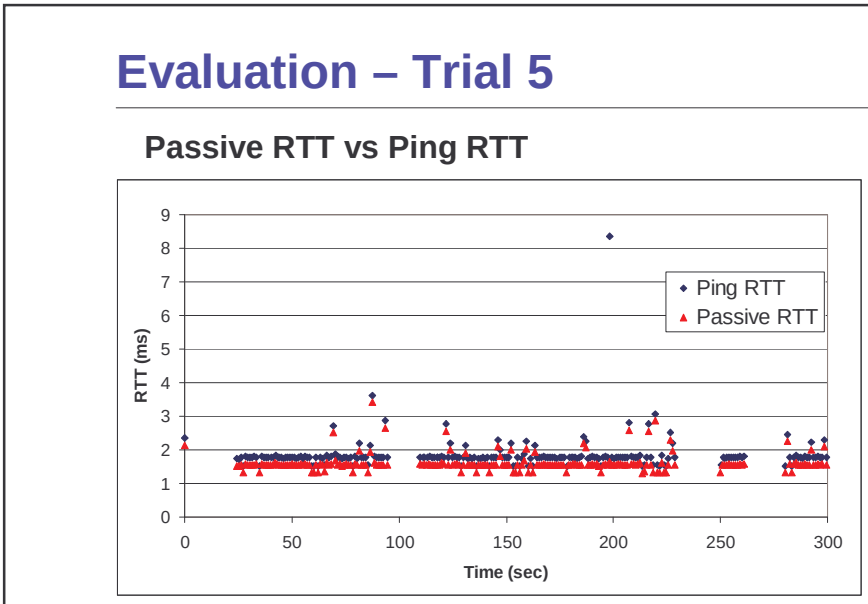
SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Evaluation – Trial 1

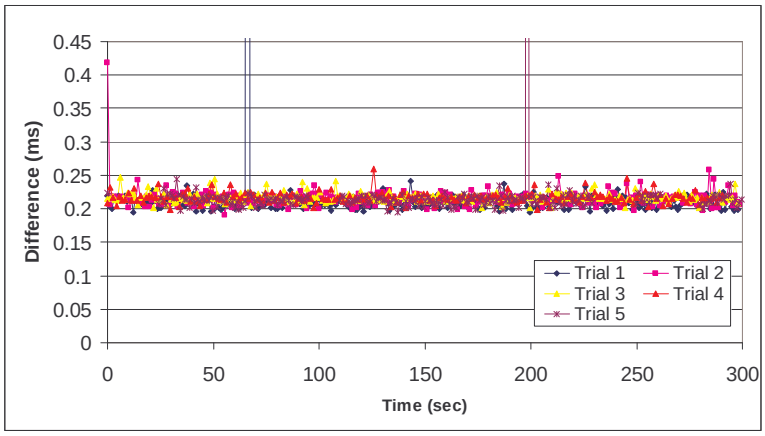


SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Evaluation – Trial 5

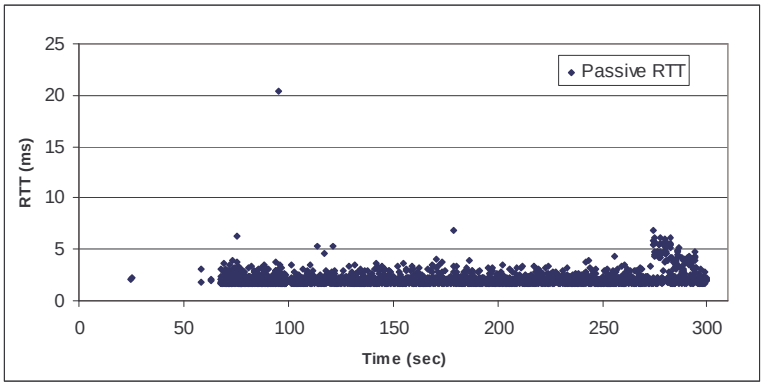


Evaluation – Delay Differences



Evaluation – Game Traffic

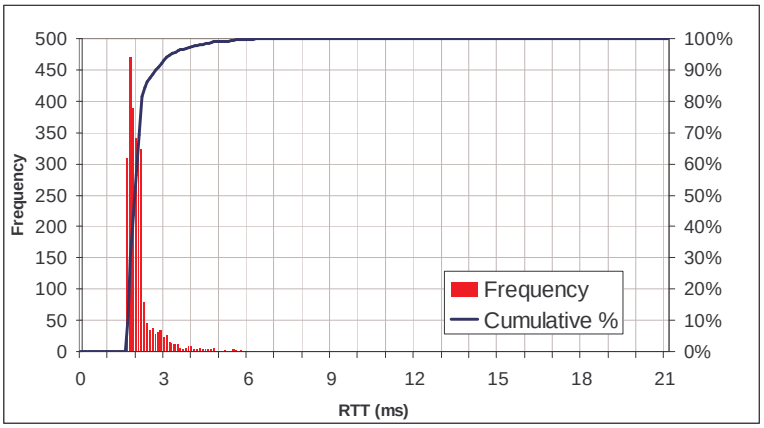
Passive RTT of ET game traffic over wireless



SWINBURNE CENTRE FOR ADVANCED INTERNET ARCHITECTURES

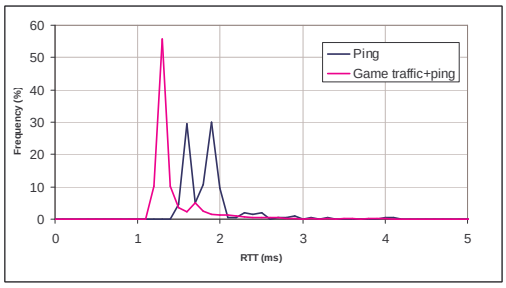
Evaluation – Game Traffic

Passive RTT of ET game traffic over wireless



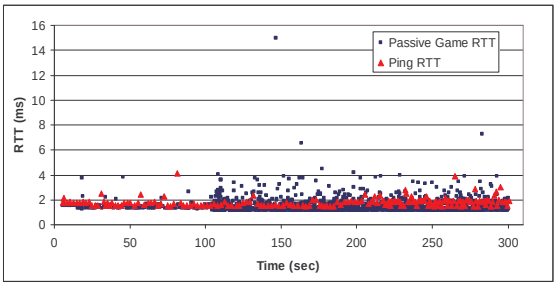
SWINBURNE CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Evaluation – Game Traffic



Distribution of Ping traffic and Game traffic including Ping traffic

Passive RTT vs Ping RTT



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)

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?

SWINBURNE CENTRE FOR ADVANCED INTERNET ARCHITECTURES