Wimeter: a novel technique for available bandwidth estimation in WLANs and its assistance to QoS provisioning

Fethi Filali

Assistant Professor

Institut Eurécom, Mobile Communications

http://www.eurecom.fr/~filali

Eurecom Seminar Series

June 1st, 2006

Motivations and Problem Statement

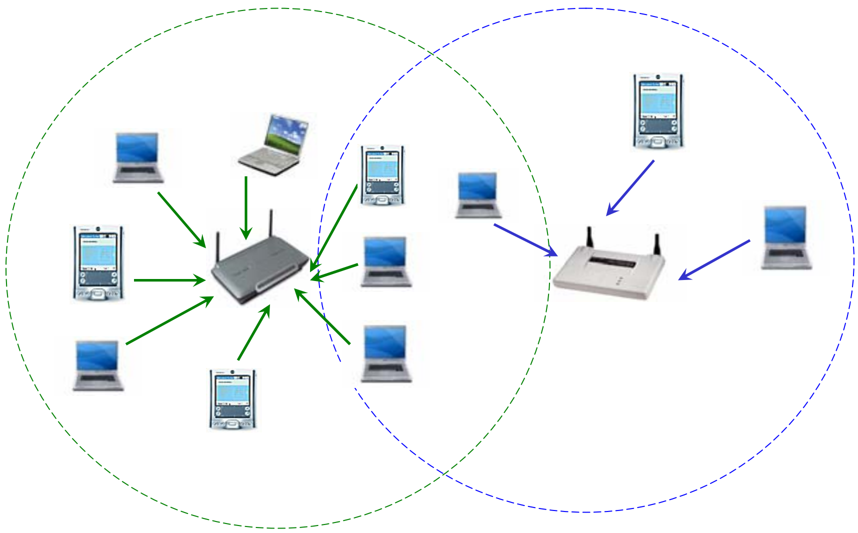
- In 802.11 each AP is assigned a channel
- 802.11 AP selection procedure: a mobile terminal selects a preferred AP based on the strength of the signal received from each AP

➡ May cause all terminals to connect to few APs: no traffic load balancing between APs

➡ Lots of contentions/collisions will occur due to CSMA/CA usage: decrease the overall goodput

➡ Handoff delays on the order of 1-2 seconds (scanning of all available channels)

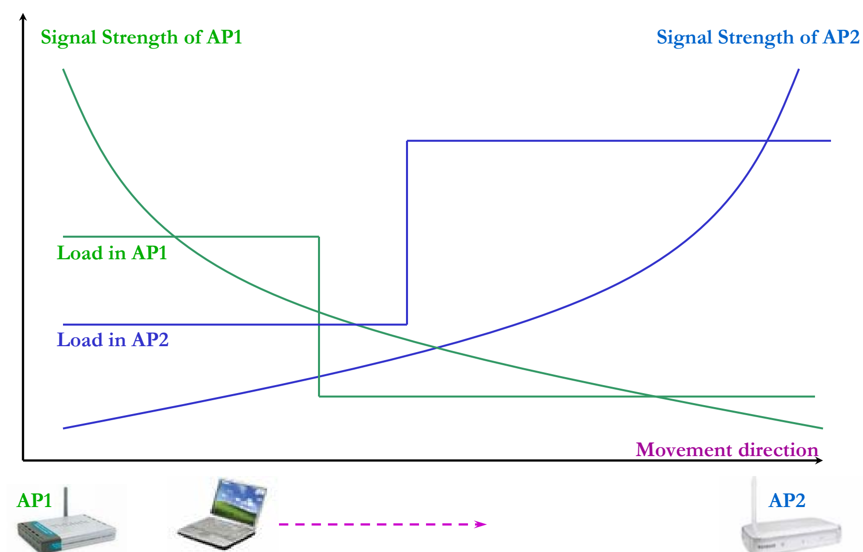
Motivations and Problem Statement



Fethi Filali

EURECOM Sophia Antipolis

Motivations and Problem Statement



Fethi Filali

EURECOM Sophia Antipolis

Objective

- Develop a smart AP selection framework taking into account
  - signal quality of APs
  - quality of service (QoS) of flows
  - achieving a traffic load balancing between APs

➡ Require the estimation of the available resources in every AP

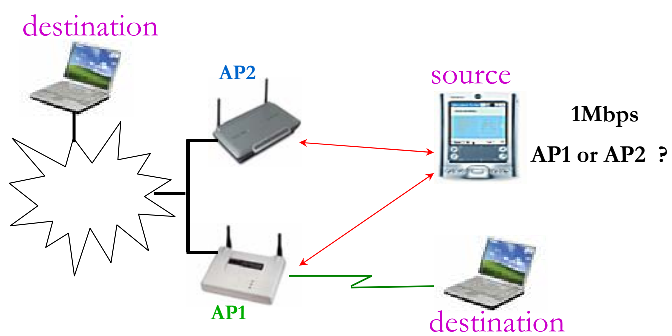
➡ Propose a novel technique to estimate the available bandwidth in a 802.11 WLAN and demonstrate its use for QoS Provisioning

Related Work and Background

- Internet proposals: several tools based on variable *packet size probing* (PVS) and *packet pair/packet train dispersion* probing (PPTD):
  - CapProbe, Pathrate, PathChar
  - PathLoad, Spruce
- They are not able to provide an accurate estimation of the bandwidth in 802.11-based networks
  - Lot of profound differences between wired and wireless networks
    - Variable capacity
    - Lost due to collisions, interferences, bad channel quality

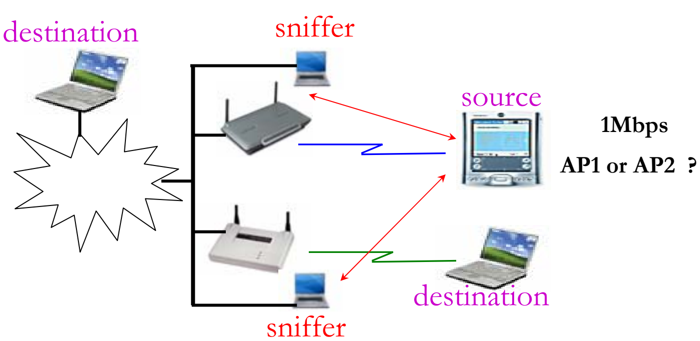
Related Work and Background

- Probing-based bandwidth estimation



Related Work and Background

- Sniffing-based bandwidth estimation



Related Work and Background

- Wireless proposals: ProbeGrap, TOPP, DietTopp, WLANTrafficProbe
- Main limitations:
  - An overhead added to estimate the available bandwidth
  - Do not take into account
    - application's packet size
    - link-layer automatic rate adaptation technique
    - bandwidth wasted because of frames lost (collisions/interferences)
  - Do not capture efficiently the backoff time
- Wimeter is a sniffing-based tool

Terminology

- The node capacity: the instantaneous channel capacity of a node in the WLAN
- The available bandwidth: the rate in Mbps at which a new flow can send traffic without affecting existing flows
- The data load: the aggregated data throughput (without transport and lower headers) of all packets transmitted in the network
- The saturation point: reached when the WLAN is saturated (available bandwidth = 0, data load = saturation throughput)

Wimeter's design challenges

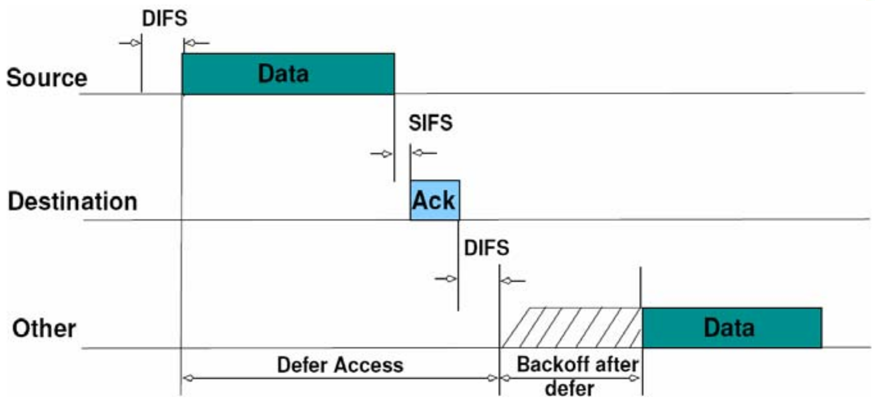
1. Capture the maximum number of frames sent in the medium. It has to be robust enough in the face of high traffic load.

2. Get the information required to estimate the available bandwidth even from encrypted frames

3. Interact with an internal mechanism that estimates the average backoff delay and the portion of the bandwidth wasted due to collisions

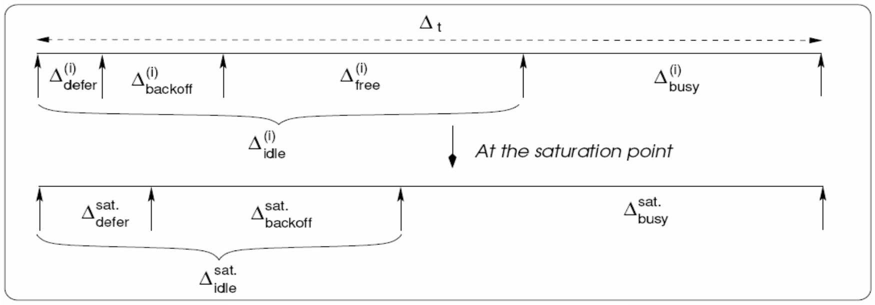
4. Avoid generating packets in the medium to have an accurate estimation

DCF Operations



- Internet bandwidth estimation techniques cannot capture the DCF operations (backoff and defer times, collision rate, etc.)

Computing the available bandwidth



$$\Delta_t = \Delta^{(i)}_{busy} + \Delta^{(i)}_{idle}$$

- The measurement period of time should be
  - larger enough to reduce the complexity of the process of analyzing the captured frames
  - small enough to have an up-to-date knowledge of the instantaneous available bandwidth

Computing the available bandwidth

$$\Delta_t = \Delta^{(i)}_{busy} + \Delta^{(i)}_{free} + \Delta^{(i)}_{defer} + \Delta^{(i)}_{backoff}$$

- At the saturation point:

$$\Delta_t = \Delta^{sat.}_{busy} + \Delta^{sat.}_{defer} + \Delta^{sat.}_{backoff}$$

- The estimated available bandwidth:

$$B^{(i)}_{free} = \left(\frac{\Delta^{sat.}_{busy} - \Delta^{(i)}_{busy}}{\Delta_t} - x^{(sat.)}_{coll.} \right) \cdot C^{sender}_{data}$$

Computing the available bandwidth

$$\Delta^{sat.}_{busy} = \Delta^{(i)}_{busy} + \Delta^{(i)}_{free} - \left(\Delta^{sat.}_{backoff} - \Delta^{(i)}_{backoff}\right) - \left(\Delta^{sat.}_{defer} - \Delta^{(i)}_{defer}\right)$$

- The number of frames allowed to be sent to reach the saturation point

$$n^{(i)}_{sat.} = \frac{\Delta^{sat.}_{busy} - \Delta^{(i)}_{busy}}{T_{frame}}$$

$$T_{frame} = \frac{S_{data}}{C^{sender}_{data}} + \frac{H_{phy}}{C^{sender}_{basic}} + \frac{S_{ack}}{C^{receiver}_{data}} + \frac{H_{phy}}{C^{receiver}_{basic}} + T_{RTS/CTS}$$

Computing the available bandwidth

$$\Delta^{sat.}_{backoff} = \Delta^{(i)}_{backoff} + \sum_{k=1}^{k=n^{(i)}_{sat}} \Delta^{sat.,k}_{backoff}$$

$$\Delta^{sat.}_{backoff} \simeq \Delta^{(i)}_{backoff} + \frac{\Delta^{sat.}_{busy} - \Delta^{(i)}_{busy}}{T_{frame}} \cdot \overline{\Delta^{pkt}_{backoff}}$$

$$\Delta^{sat.}_{defer} = \Delta^{(i)}_{defer} + \frac{\Delta^{sat.}_{busy} - \Delta^{(i)}_{busy}}{T_{frame}} \cdot \Delta^{pkt}_{defer}$$

$$\Delta^{sat.}_{busy} = \Delta^{(i)}_{busy} + \Delta^{(i)}_{free} - \frac{\Delta^{sat.}_{busy} - \Delta^{(i)}_{busy}}{T_{frame}} \cdot \left(\overline{\Delta^{pkt}_{backoff}} + \Delta^{pkt}_{defer}\right)$$

Computing the available bandwidth

$$\Delta_{busy}^{sat.} = \Delta_{busy}^{(i)} + \frac{T_{frame}}{T_{frame} + \overline{\Delta_{backoff}^{pkt}} + \Delta_{defer}^{pkt}} \cdot \Delta_{free}^{(i)}$$

- The estimated available bandwidth:

$$B_{free}^{(i)} = \left(\frac{T_{frame}}{T_{frame} + \overline{\Delta_{backoff}^{pkt}} + \Delta_{defer}^{pkt}} \cdot \frac{\Delta_{free}^{(i)}}{\Delta_t} - x_{coll.}^{(sat.)} \right) \cdot C_{data}^{sender}$$

Computing the available bandwidth

$$\Delta_{free}^{(i)} = \Delta_{idle}^{(i)} - \Delta_{defer}^{(i)} - \Delta_{backoff}^{(i)}$$

$$\begin{aligned}
\Delta_{idle}^{(i)} &= \Delta_t - \Delta_{busy}^{(i)} \\
&= \Delta_t - \frac{NetworkLoad^{(i)}}{ChannelCapacity} \cdot \Delta_t \\
&= \Delta_t \cdot \left(1 - \frac{NetworkLoad^{(i)}}{ChannelCapacity}\right)
\end{aligned}$$

$$\begin{aligned}
\Delta_{defer}^{(i)} &= ((N_{ACK}^{(i)} + N_{RTS}^{(i)} + N_{CTS}^{(i)}) \cdot SIFS \\
&+ N_{DATA}^{(i)} \cdot DIFS) \cdot \text{aSlotTime.}
\end{aligned}$$

Computing the available bandwidth

- for unicast data frames: $\Delta^{pkt}_{defer} = (3.SIFS + DIFS) \cdot \text{aSlotTime}$: when using RTS/CTS technique, and $\Delta^{pkt}_{defer} = (DIFS + SIFS) \cdot \text{aSlotTime}$: in case of not using this technique.
- for multicast and broadcast data frames: $\Delta^{pkt}_{defer} = SIFS \cdot \text{aSlotTime}$ regardless of the use of RTS/CTS mechanism.

Computing the backoff time

- Backoff advertisement: ask the MAC layer to announce the randomly backoff time used
  - Requires the modification of the 802.11 MAC header by adding a new field which should contain the backoff time
  - Not compliant with the 802.11 standard
- Measuring the backoff time at the access point: thanks to 802.11 protocol, all stations would experience the same average backoff delay
  - Measuring this parameter at the AP will be sufficient
  - Implies a modification of the 802.11 driver at the AP in order to periodically compute the average backoff time and report to wimeter the obtained value
  - This is not compliant with the design challenges

Computing the backoff time

- Analytic-based estimation: use of the Bianchi model-like analytic framework: Markov chain-based modeling

$$\Delta_{backoff}^{(i)} = \sum_{k=1}^{k=n_{data}^{(i)}} \Delta_{backoff}^{i,k}$$

$$\begin{aligned}
\overline{\Delta_{backoff}^{pkt}} &= \overline{W} = \sum_{j=0}^{m} \pi_{j,0} \frac{W_j - 1}{2} \\
&= \pi_{0,0}\frac{W_0 - 1}{2} + \sum_{j=1}^{m-1} p^j \pi_{0,0} \frac{2^j W_0 - 1}{2} \\
&+ \frac{p^m}{1-p}\pi_{0,0}\frac{2^m W_0 - 1}{2}
\end{aligned}$$

Computing the backoff time

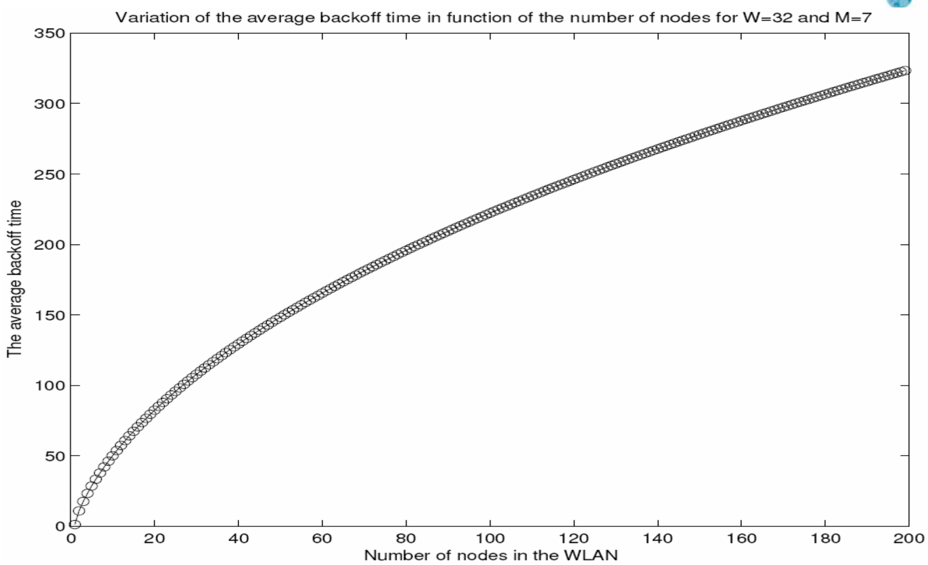
$$\begin{aligned}
\overline{\Delta_{backoff}^{pkt}} &= \frac{1}{(1-2p)(W_0+1) + pW_0(1-(2p)^m)} \\
&\cdot ((1-2p)(1-p)(W_0-1) \\
&+ 2pW_0(1-(2p)^{m-1})(1-p) \\
&- p(1-p^{m-1})(1-2p) \\
&+ p^m(2^m W_0 - 1)(1-2p))
\end{aligned}$$

- *p*: unsuccessful transmission probability during a time-slot

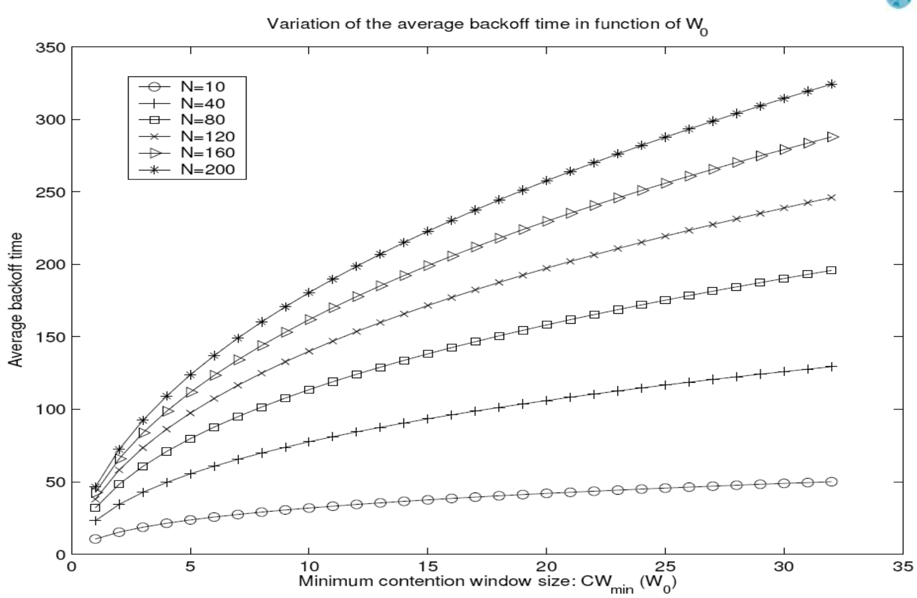
$$p = 1 - (1-\tau)^{n-1}$$

- τ: transmission probability
- m: maximum number of retransmissions
- W_0: minimum contention window size
- n: number of active mobile stations

Computing the backoff time



Computing the backoff time



Computing the backoff time

- Implementing the proposed analytic technique:

$\Delta_{busy}^{(i,k)}$
- the portion of the busy time caused by the frames transmitted by the station *k*

n_i
- the number of active mobile stations

$$\tau = \frac{1}{n_i}\sum_{k=1}^{k=n_i}\frac{\Delta_{busy}^{(i,k)}}{\Delta_t} = \frac{1}{n_i}\frac{\Delta_{busy}^{(i)}}{\Delta_t}$$

- Apply an Exponential Weighted Moving Average (EWMA)

$$\tau = (1-\alpha)\tau + \alpha\tau^{(i)}$$

Computing the collision rate

- Implementing the proposed analytic technique

$$p_{tr} = 1-(1-\tau)^n$$

- the probability that there is at least one transmission in a timeslot

$$p_s = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n}$$

- the probability of a successful transmission

$$T_{coll.}^{(i,sat.)} = \sum_{k=1}^{k=n_{sat.}^{(i)}} p_{tr}(1-p_s)T_{coll.}$$

$$T_{coll.} = DIFS + \overline{\Delta_{backoff}^{pkt}} + \frac{H_{phy}}{C_{basic}^{sender}} + \frac{S_{data}}{C_{sender}}$$

Computing the collision rate

$T^{(i)}_{coll.}$: the time wasted due to collisions during the measurement period *i*

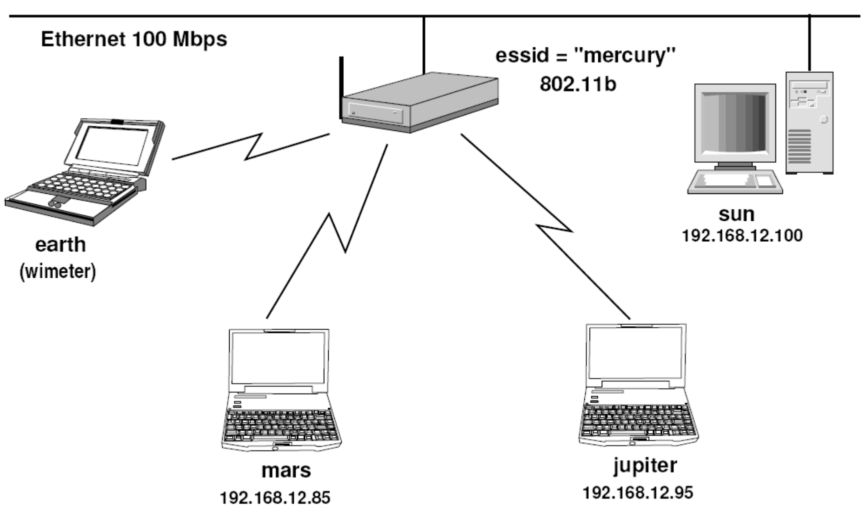
- could be measured on real-time by the machine

$$T^{(sat.)}_{coll.} = T^{(i)}_{coll.} + T^{(i,sat.)}_{coll.}$$

- The collision rate is:

$$x^{(sat.)}_{coll.} = \frac{T^{(sat.)}_{coll.}}{\Delta_t} = \frac{T^{(i)}_{coll.} + T^{(i,sat.)}_{coll.}}{\Delta_t}$$

Real-experimentation - testbed

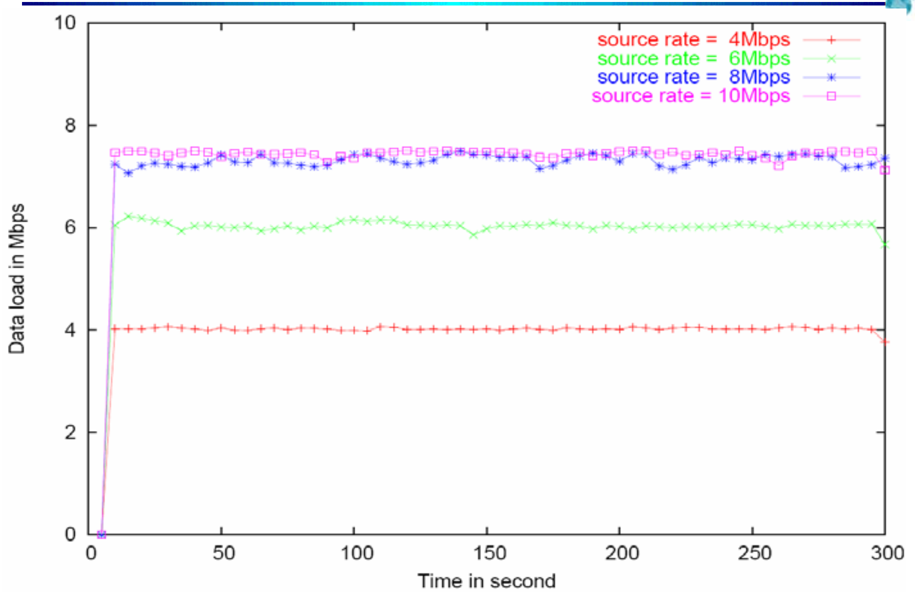


Real-experimentation - configuration

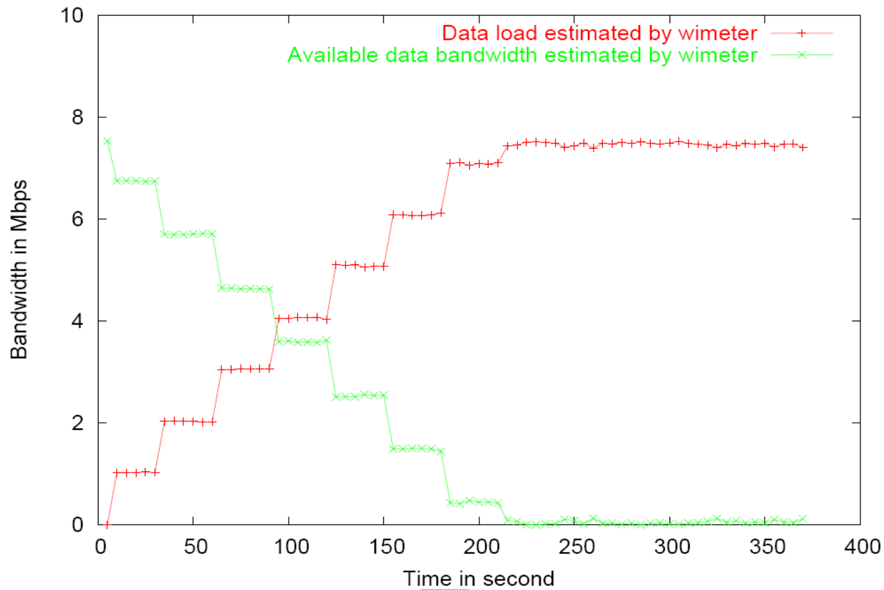
- Wimeter was implemented in C for Linux OS
- Wifi cards: Netgear WAG 511 (MADWIFI driver)
- Access point: Cisco Aironet
- Traffic generated using MGEN tool

Parameter's name	Parameter's value
DIFS	$50\mu s$
SIFS	$10\mu s$
aSlotTime	$20\mu s$
$CW_{min} = W_0$	31
BasicRate	1 Mbps
LinkCapacity	11 Mbps
PLCP header size	24 bytes
Frame control size	32 bytes
MAC ACK size	14 bytes

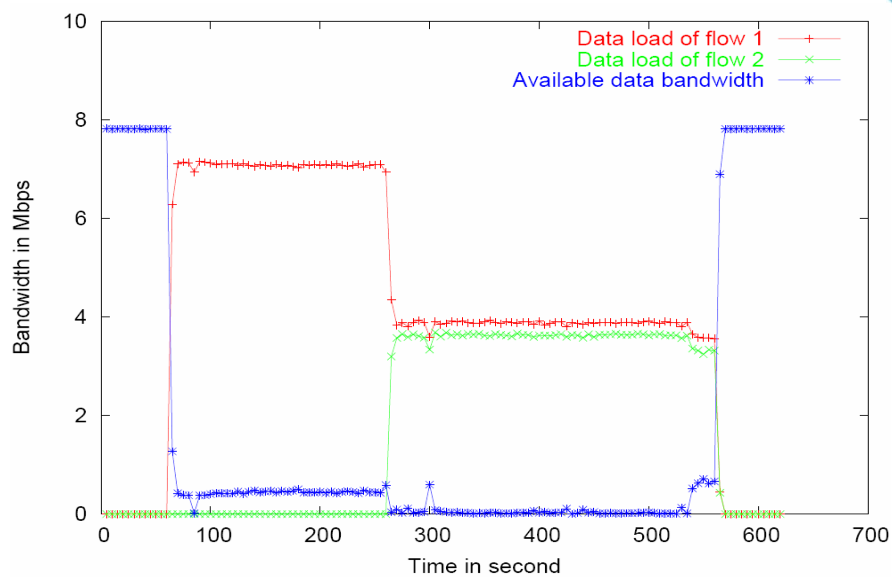
Real-experimentation - results



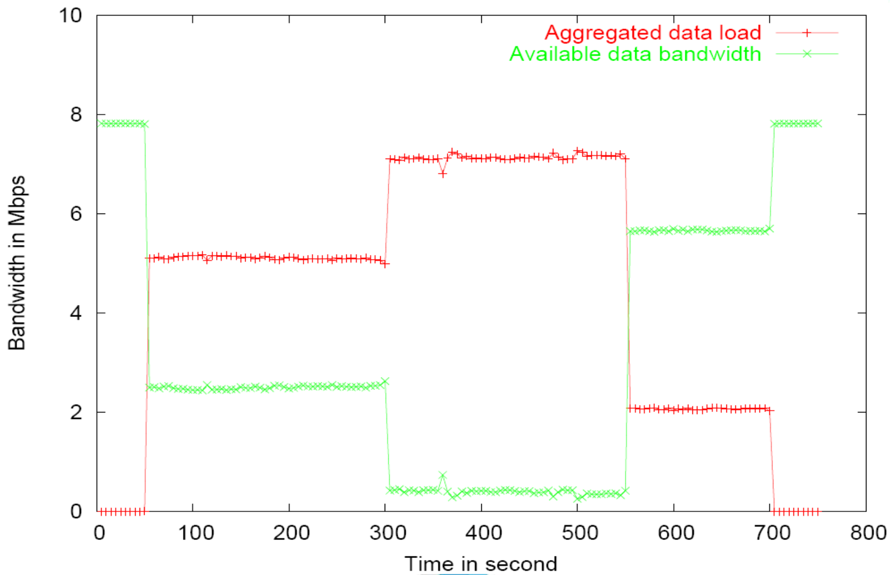
Real-experimentation - results



Real-experimentation - results

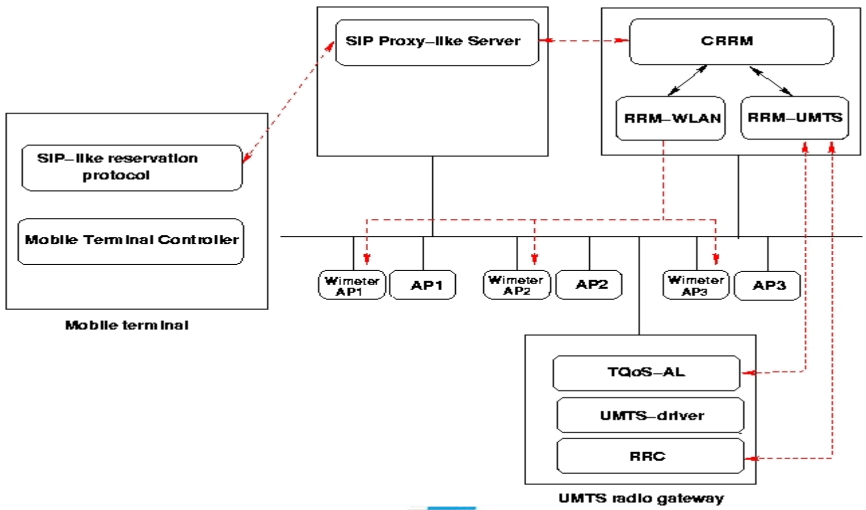


Real-experimentation - results

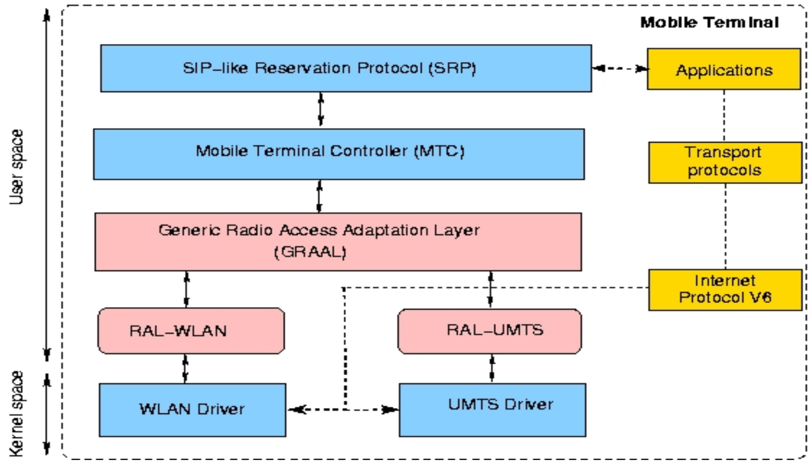


Integration of wimeter in Eurecom Platform

- Eurecom's software-radio heterogeneous wireless platform

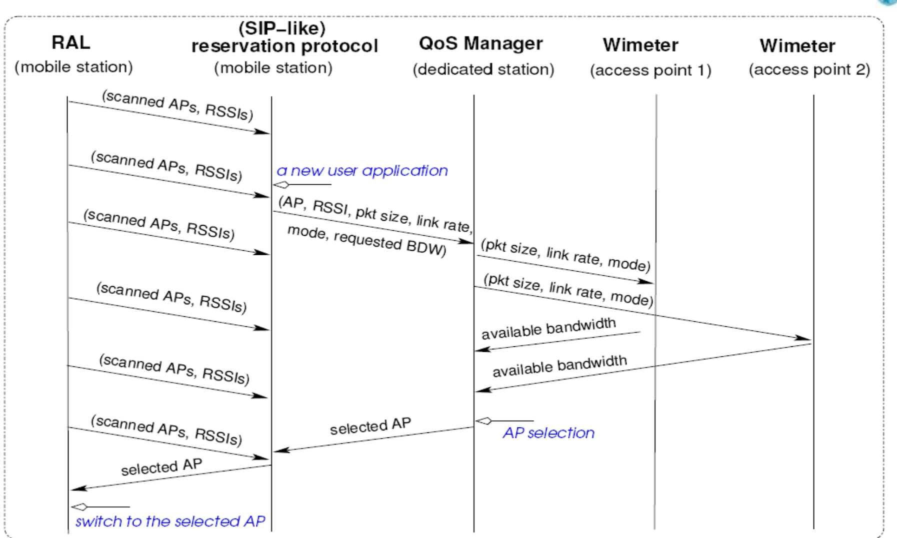


Mobile terminal components



- RNRT: rhodos, cosinus
- IST FP6: Daidalos(I and II), E2R (I and II)

Wimeter-aided WLAN CAC Framework



Possible extensions

- To 802.11e WLANs:
  - estimated available bandwidth should be estimated for each traffic category (TC)
  - use a 802.11e analytic model for estimating the average backoff time and the collision rate for each for each TC
  - DLP may induce some problems !
- To 802.11-based MANETs:
  - run wimeter in each mobile station
  - the estimation of the collision rate has to be adapted to MANETs.
  - take into account the presence of the exposed problem
  - the number of active stations is not the total number of nodes in the MANET but only those which will be affected by the one-hop bandwidth reservation

Conclusion

- Propose a new framework to efficiently balancing the load between available access points
- Design a new sniffing-based algorithm for available bandwidth estimation in 802.11-based wireless networks: wimeter
- Wimeter interacts with other modules in order to provide the most suitable access point for a given connection
- We showed that wimeter is able to perfectly capture the data packets and estimated the load for data, control, and management packets
- Wimeter is able to compute efficiently the available bandwidth which depends on the packet size and the link-layer rate of the sender and the receiver