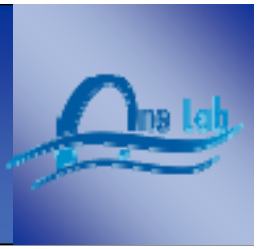
Università di Pisa
Dipartimento Ing. dell'Informazione



Wireless link emulation in OneLab

(work in progress)

G. Cecchetti, M. Carbone, L. Rizzo, F. Checconi, A. L. Ruscelli
Università di Pisa – Scuola Superiore S. Anna
Italy

July 11-12, 2007
ROADS – Warsaw, Poland

What is OneLab

- **OneLab is an open networking laboratory integrating, testing, validating and demonstrating new fixed and wireless networking technologies in real world settings and production environments**

Goals

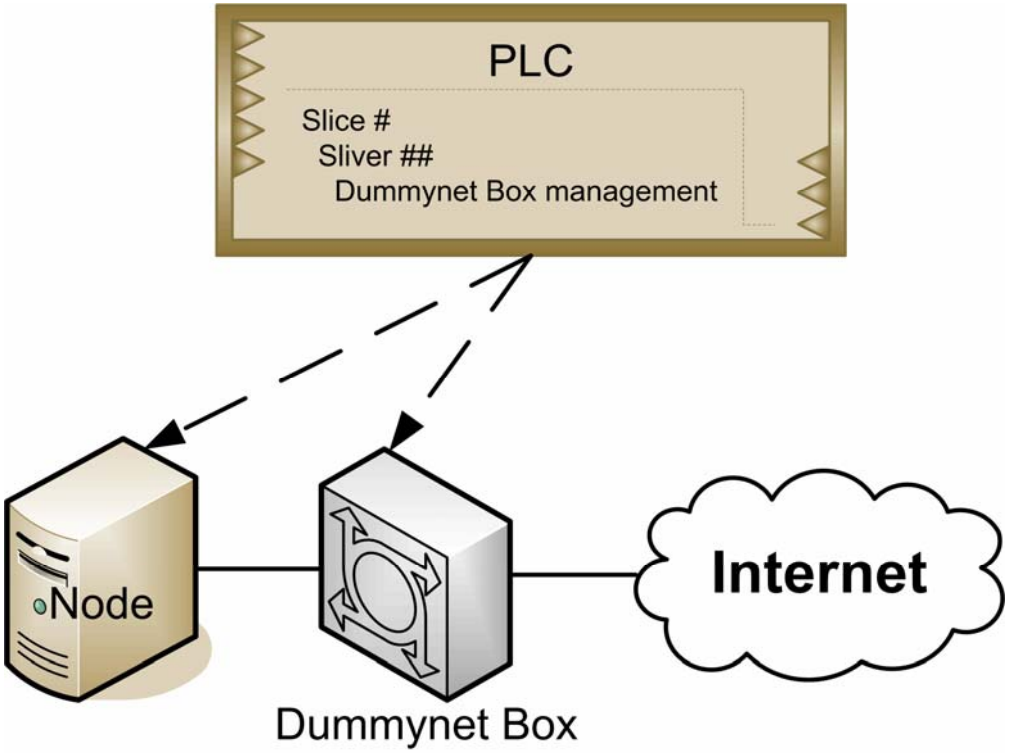
- **Our goal** is to *extend* OneLab with an emulation component which can reproduce the effects of a wireless link on live traffic
  - In this way we can run repeatable experiments under controlled conditions of wireless network
  - Different experiments can run concurrently with different (emulated) network conditions, if they need to

Problems

1. *Modification* to existing **OneLab architecture**
   - Adding new entities (emulation boxes)
2. *Change* **PLC management** interface to link nodes and emulation boxes
3. *Definition* of a simple yet flexible **API** that can be used at various level (configuration scripts or running experiments) to modify the features of the emulated link
4. *Update* the **dummynet** code to emulate the wireless link

Main Idea

- Make the network link to a OneLab node behave as a wireless link



Main things to do

- PLC extensions
- A model for wireless dummynet

PLC extensions

- The addition of the emulation box requires **instructing the PLC software to store information about the new component**, and its connection to the nodes by:
  - Adding suitable fields to the database tables on the central site
  - Update the management software to handle this information
- The PLC software is also in **charge of producing software images for the emulation box**, same as it is done for other OneLab nodes.
- **Configuration and authentication information** is also distributed to the emulation boxes using the existing mechanism and APIs.

PLC modifications

- We **need to add the following information** on the central site's database:
  - ***node capabilities***, i.e. indicating the ability to choose among different network links and configure them. This is a modification of general interest;
- We also **need to modify the central node** so it can send *configuration information* to a node or to its link controller (the dummynet box, etc. . .) before the slice is made available for an experiment

Node specific modifications

- A "`netconfig`" program to configure the network link
- A *configuration file* where information about the DummyBox *reachability* are stored

Specific software on DummyBox

- The modified *dummynet* code
- A *ssh* server listening for configuration
- Some python programs that periodically check for updates (e.g. the emulation configuration info from the central site to the middle box when the sliver is activated)

The Approach to model a Wireless Dummynet

- Study a model for a wireless dummynet
  - Protocol analysis
  - Evaluation of wireless parameters
- Introduce a middle box (based on modified dummynet code) to emulate the link

What we do not want model for wireless Dummynet

- Modeling all the features of the MAC layer is outside of the scope of the platform due to:
  - Artifacts
  - Uncertainty of the rest of the network

- These act on a coarser timescale!

  → We do not want to model individual events on a microsecond timescale

  → We only look at the aggregate timescale

What we want model for wireless Dummynet

- Initially, emulate 802.11 DCF mode
- Our focus is on the effects of interest for an experiment running on OneLab node such as:
  - ***min/max physical transmission rates*** (obviously);
  - ***S/N ratio***, as it affects the rate selection algorithm and the error rate of the link;
  - ***numbers of active clients***, as it affects the delay in accessing the media, and also the available share of bandwidth;
  - ***trashing effects***, related to access point associations and collisions;

Protocol Analysis

- From the study of IEEE 802.11 protocol we can infer an initial set of high level parameters that affect individual packet transmissions:
  - PROTOCOL_OVERHEAD is the **time while the channel is not used for actual data transmission** due to arbitration intervals, preamble and mandatory protocol fields (100-600us)
  - DATA_RATE is the **raw bit rate for payload** transmissions as configured by the user and the rate adaptation mechanism

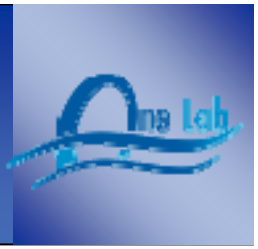
Evaluation of Wireless parameters for Dummynet

- We model effects as number of clients and SNR with two additional parameters:
  - CONTENTION_RATE is the **probability that acquiring the channel fails because of contention**, causing the station to wait until the winner of the arbitration completes its transmission (this depends on the n. of the active clients and influences the tx delay)
  - TX_FAILURE_RATE is the **probability that a packet transmission fails** and so a *retransmission is required*

Main work about Wireless Dummnet

- Dummynet already implements DATA_RATE
- **It needs to be extended to account for the extra time needed to complete a packet transmission** (due to PROTOCOL_OVERHEAD)
- To evaluate the delay due to CONTENTION_RATE we **analyze CRAWDAD trace data to highlight the influence of active nodes number**
- To compute the delay due to TX_FAILURE_RATE we **measure the delays between the data tx and their corresponding ACKs both in case of successful tx and in case of re-tx**.
- *Other analysis is in progress!*

How to run an experiment



- When a user has to start an experiment first of all it needs to configure the DummyBox, or select a specific network link, etc...
- It will simply launch a one-line "`netconfig ...`" program with all the parameter required by thespecific network link
- In turn, "`netconfig`" will call the appropriate configuration program (typically a "*setuserid*" program to configure a local interface)

An example experiment: *ping performance*

- We want to make an experiment to see how a “ping” depends of the number of active stations present in a wireless communication.

- **First step**: *set-up the emulated link*:

```
./netconfig.sh emu_802 5000 5 1Mbps 38dB
```

An example experiment: *ping performance*

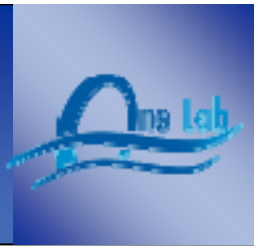
- After this command we can see that the link is configured:

```
./netconfig.sh emu_802 5000 5 1Mbps 38dB
The 802.11 emulation link is configured with
5 active stations, 1Mbps of bandwidth and a
SNR of 38dB on 131.114.9.236:5000
```

- **Second step**: now we can *start the program* specifying the use of the previously configured link:

```
./hping2 -s 5000 -c 1 onelab7.iet.unipi.it
```

An example experiment: *ping performance*



- And this is the result:

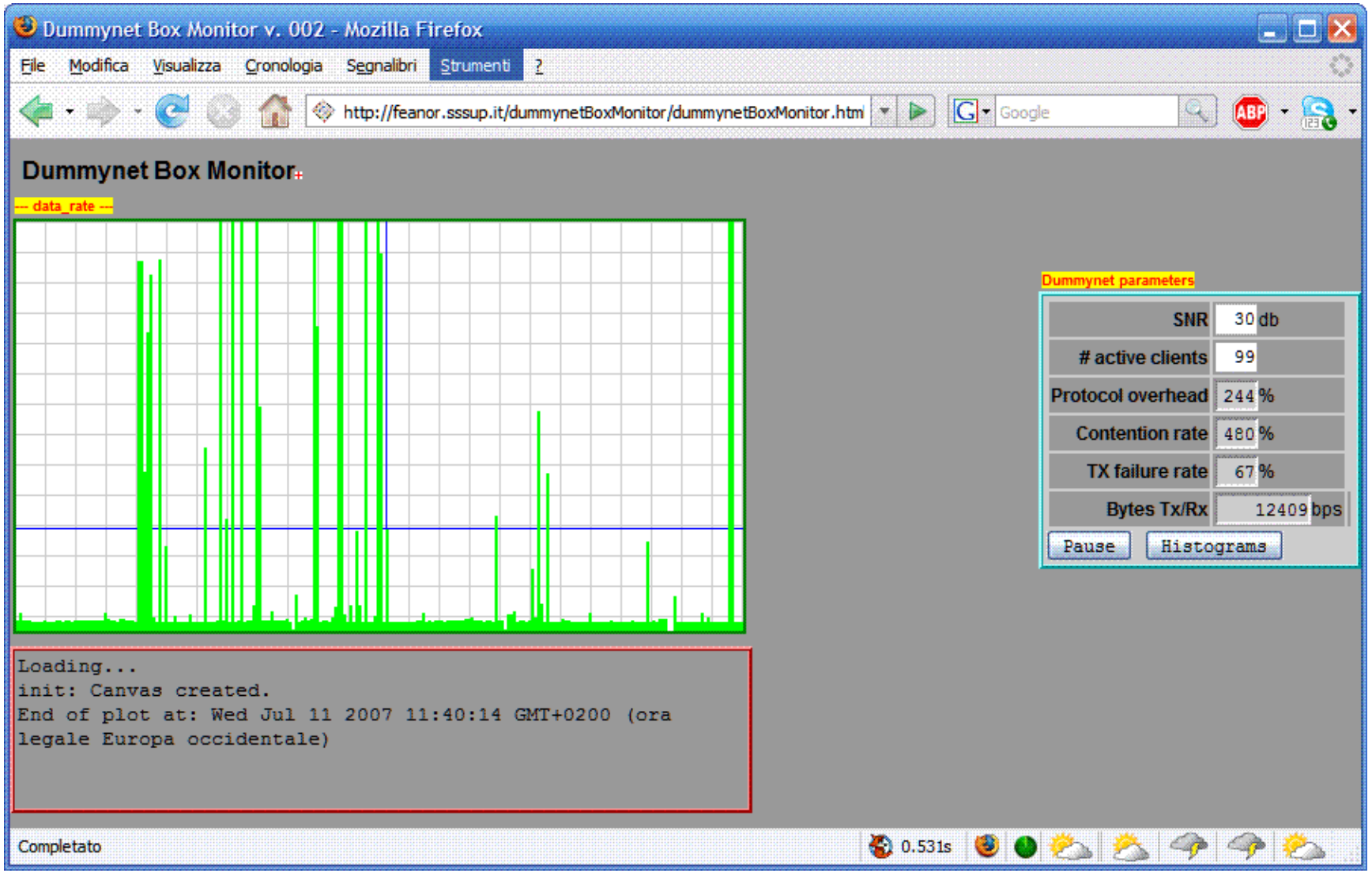
```
./hping2 -s 5000 -c 1 onelab7.iet.unipi.it
HPING onelab7.iet.unipi.it (em0 131.114.9.134):\
    NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=131.114.9.134 ttl=64 DF id=266 sport=0\
    flags=RA seq=0 win=0 rtt=0.2 ms

--- onelab7.iet.unipi.it hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.2/0.2/0.2 ms
```

Conclusions

- We are developing an **extension to Dummynet for OneLab network** that could have great number of applications and serve well the OneLab/PlanetLab community to generate comparable and reproducible results
- This extension require some update of PLC software
- The *study of wireless model* to be used for Dummynet is a critical point:
  - initial parameters proposed have to be checked
  - a deep comparison between the emulated behavior and a real system is needed to fully understand how realistic the emulator will be
- The emulation component have to be realistic-enough emulation without adding too much complexity

Questions ?


Dummynet Box Monitor v. 002 - Mozilla Firefox
File Modifica Visualizza Cronologia Segnalibri Strumenti ?
http://feanor.sssup.it/dummynetBoxMonitor/dummynetBoxMonitor.htm
Google
Dummynet Box Monitor
-- data_rate --
Dummynet parameters
SNR 30 db
active clients 99
Protocol overhead 244 %
Contention rate 480 %
TX failure rate 67 %
Bytes Tx/Rx 12409 bps
Pause
Histograms
Loading...
init: Canvas created.
End of plot at: Wed Jul 11 2007 11:40:14 GMT+0200 (ora legale Europa occidentale)
Completato
0.531s