The OneLab Project

Thierry Parmentelat, INRIA

PlanetLab Federation Workshop
Palo Alto, May 11th, 2006
OneLab in a nutshell

- EU funded
- Timeframe: Sept. 1st 2006, for 2 years

- 10-parties consortium [Ericsson has left]
  
  **Academic**  UPMC/LIP6, INRIA,
  Universities: Madrid, Louvain, Napoli, Pisa

  **Industrial**  Intel Research Cambridge, France Telecom,
  Alcatel Italia, Telekomunikacja Polska

- Objectives
  - Operate PlanetLab Europe - PLE
  - New features: wireless, monitoring, emulation
Operating PLE

Why operating a PLE?

- First perceived as a goal per se
- Then as a means to deploy our features
- Now both objectives have similar visibility

Some figures

142/67 in Germany, UK, Switzerland, Italy, France, Poland, Spain, Sweden, Greece, Netherlands, Finland

35/10 OneLab partners PlanetLab only

?? Find balance/incentives between MA and SA
Expectations from federation

- Required for running PLE

- Dissemination strategy: next step after Private PlanetLab
  - Bridge the gap between “strictly private” and “consortium-wide” approaches

- Provide a framework for creating incentives
  - e.g. UMTS-centric SA
WiFi in PlanetLab nodes

- Set up a PlanetLab node
  - As a regular WiFi client - wide HW spectrum
  - As an Access-Point - madwifi-ng specific (exp.)

- Node-side only so far, still missing:
  - DB / UI (WiFi settings, multiple interfaces)

    - specification required (remain user-friendly)
    - DB a legacy-driven component, hard to evaluate impact

- No operational private PLC anymore
Changes required so far

- **build system**
  - **Purpose**: attempt to ease upgrades
  - **Means**: build from various types of sources
    - cvs, svn, cached tarballs, srpm
  - with home-maintained patches
  - connection with pl_box, to be updated with myplc

- **kernel**
  - **Purpose**: add WiFi capabilities
  - **Means**: change config
  - and add external driver modules (madwifi-ng)
Assumptions for the boot sequence:
- Node typically has multiple interfaces
- Stage 1 (bootCD): uses one interface
- Stage 2 (bootmanager): may involve several interfaces

So far
- Single interface nodes use the same for both stages

Other setups could be envisioned
- E.g. a 'service' wired interface
- Reserved for boot, updates & operations
- Hidden to users
Changes required (end)

- **bootCD**
  - added patched tools (initscripts, pcmcia-CS)
  - added regular external tools, (hotplug)
  - (patches mostly from more recent versions)
  - HW (PCI) initialization revisited
  - WLAN settings in plnode.txt
  - interface name exported
  - various modprobe.conf tunings
  - support for debugging breakpoints
  - scripting node-specific CD creation

- **bootmanager**
  - network configuration reworked
  - blacklisted erratic modules for kexec
  - support for debugging breakpoints
How to handle features/flavors?

- The MA is responsible for its operations
  - Not likely to accept third-party changes without review

- OK, but should this code remain stuck in my lab?

- Embed heterogeneity into the framework (node/server)
Lessons

• Development & testing really slow

• myplc a great move

• Early stage: local svn repo → PLC’s cvs & branches
• Framework to deal with heterogeneity
• Better visibility on evolutions & deployment practices
• Automated test framework - even minimalist - would help
Remaining issues

- server-side (DB...)
- Other types of multiple interfaces (multihoming)
- (ipv6)

- Wireless links virtualization for effective sharing
- Mobility
- → reservation schemes
• Pave the way towards federation
• Improve a collaborative development process