**PlanetLab: A Distributed Test Lab for Planetary Scale Network Services**

**Opportunities**
- Emerging “Killer Apps”:
  - CDNs and P2P networks are first examples
  - Application spreads itself over the Internet
- Vibrant Research Community:
  - Distributed Hash Tables: Chord, CAN, Tapestry, Pastry
  - Distributed Storage: Oceanstore, Mmemosyne, Past
  - Lack of viable testbed for ideas

**Synopsis**
- Open, planetary-scale research and experimental facility
- Dual-role: Research Testbed AND Deployment Platform
- >1000 viewpoints (compute nodes) on the internet
- 10-100 resource-rich sites at network crossroads
- Analysis of infrastructure enhancements,
- Experimentation with new applications and services
- Typical use involves a slice across substantial subset of nodes

**What will PlanetLab enable?**
- The open infrastructure that enables the next generation of planetary scale services to be invented
- Post-cluster, post-yahoo, post-CDN, post-P2P, ...
- Potentially, the foundation on which the next Internet can emerge
- A different kind of testbed
- Focus and Mobilize the Network / Systems Research Community
- Position Intel to lead in the emerging Internet

**New Ideas / Opportunities**
- Service-centric Virtualization
  - Re-emergence of Virtual Machine technology (VMWare…)
  - Sandboxing to provide virtual servers (Ensim, UML, Vservers)
  - Network Services require fundamentally simpler virtual machines, making them more scalable (more VMs per PM), focussed on service requirements.
  - Instrumentation and Management become further virtualized “slices”
- Restricted API => Simple Machine Monitor
  - Very simple monitor => push complexity up to where it can be managed
  - Ultimately one can only make very tiny machine monitor truly secure
  - SILK effort (Princeton) captures most valuable part of ANets nodeOS in Linux kernel modules
  - API should self-virtualize: deploy the next PlanetLab within the current one
- Planned Obsolescence of Building Block Services
  - Community-driven service definition and development
  - Service components on node run in just another VM
  - Team develops bootstrap ‘global’ services

**The Hard Problems**
- “Slice-ability”: multiple experimental services sharing many nodes
- Security and Integrity
- Management
- Building Blocks and Primitives
- Instrumentation

**People and Organization**
- Project Owner: Timothy Roscoe (acting)
- Hans Mulder (sponsor)
- David Culler
- Larry Peterson
- Tom Anderson
- Milan Milenkovic
- Earl Hines

**Overall Timeline**
- Phase 0: 2002
  - Mission Control System Rev 0
  - Centralized Mgmt Services
  - Thin, event-driven monitor
  - Host new services directly
  - Host phase 1 as virtual OS
  - Replace bootstrap services
- Phase 1: 2003
  - Secure thinix
  - Native thinix Mgmt Services
  - Bootstraps hosted in VMs
  - Outsource operational support
- Phase 2: 2004
  - Planck API in Linux
  - Transition Open Mgmt to Community
  - Broader Consortium
  - “Slice-ability”
- Phase 3: 2005
  - Extend to 1000+ nodes
  - Distributed deployment of self-serviced infrastructure
  - Transition Open Mgmt to Community
  - Broader Consortium
  - “Slice-ability”

**Project Strategy**
- What will PlanetLab enable?
  - The open infrastructure that enables the next generation of planetary scale services to be invented
  - Post-cluster, post-yahoo, post-CDN, post-P2P, ...
  - Potentially, the foundation on which the next Internet can emerge
  - A different kind of testbed
  - Focus and Mobilize the Network / Systems Research Community
  - Position Intel to lead in the emerging Internet

**Seed Research & Design Community**
- MIT
- Utah
- Duke
- UCSD
- Rice
- Berkeley
- Washington
- Princeton

**Larger academic, industry, government non-profit organization**
- CMU
- Pittsburg

**Core Engineering & Design Team**
- More academic, industry, government non-profit organizations
- More comprehensive infrastructure
- More comprehensive services

**Distributed Design Team**
- More academic, industry, government non-profit organizations
- More comprehensive infrastructure
- More comprehensive services

**Minimal VM / Auth. requirements**
- Applications mostly self-sufficient
- Core team manages platform