PlanetLab

The Design Principles of PlanetLab,

Larry Peterson and Timothy Roscoe, June 2004 (updated January 2006).

PlanetLab: Evolution vs Intelligent Design in Global Network Infrastructure,

Larry Peterson

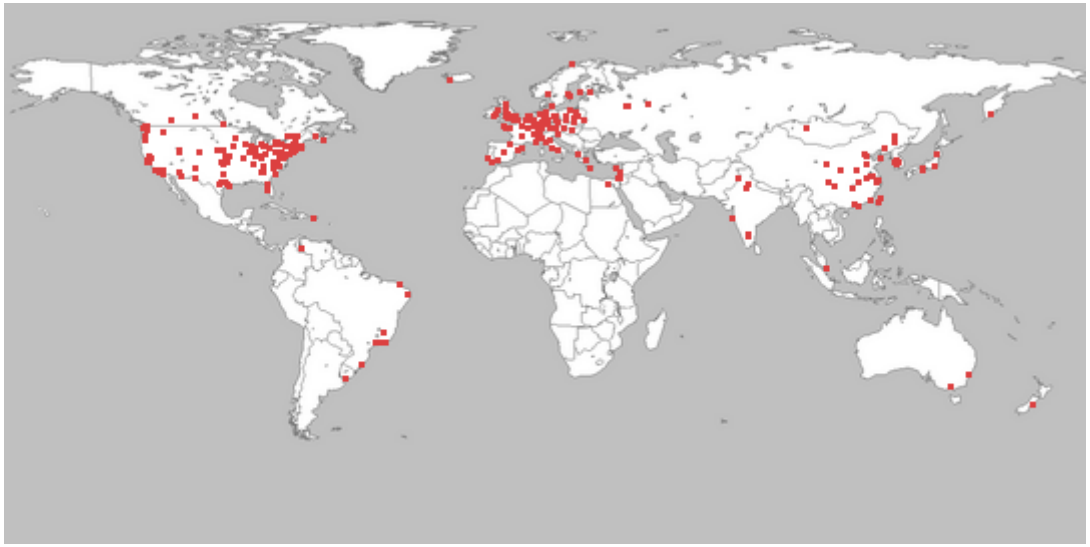
Youngbin Im
ybim@mmlab.snu.ac.kr
2007.11.14.

Contents

- Introduction to PlanetLab
- Design goals of PlanetLab
- Virtualization
- PlanetLab terminology
- Design challenges
- Design principles
  - Distributed virtualization
  - Unbundled management
  - Chain of responsibility
  - Evolution vs. clean slates
  - OS and control plane
- VNET

Introduction to PlanetLab

- PlanetLab
  - A group of computers available as a testbed for computer networking and distributed systems research. –wikipedia
  - An open platform for developing, deploying, and accessing planetary-scale services – www.planet-lab.org
  - currently consists of 825 nodes at 406 sites



Introduction to PlanetLab

- Since 2003, 1000+ researchers at academic institutions and industrial research labs have used to develop new tech for
  - distributed storage
  - network mapping
    - The study of the physical connectivity of the Internet
  - peer-to-peer systems
  - distributed hash tables
    - A distributed system that provides a lookup service similar to a hash table
  - query processing

Introduction to PlanetLab

- Content Distribution
  - CoDeeN: Princeton
  - Coral: NYU, Stanford
  - Cobweb: Cornell
- Storage & Large File Transfer
  - LOCI: Tennessee
  - CoBlitz: Princeton
- Information Plane
  - PIER: Berkeley, Intel
  - PlanetSeer: Princeton
  - iPlane: Washington
- DHT
  - Bamboo (OpenDHT): Berkeley, Intel
  - Chord (DHash): MIT
- Routing / Mobile Access
  - i3: Berkeley
  - DHARMA: UIUC
  - VINI: Princeton
- DNS
  - CoDNS: Princeton
  - CoDoNs: Cornell
- Multicast
  - End System Multicast: CMU
  - Tmesh: Michigan
- Anycast / Location Service
  - Meridian: Cornell
  - Oasis: NYU
- Internet Measurement
  - ScriptRoute: Washington, Maryland
- Pub-Sub
  - Corona: Cornell
- Email
  - ePost: Rice
- Management Services
  - Stork (environment service): Arizona
  - Emulab (provisioning service): Utah
  - Sirius (brokerage service): Georgia
  - CoMon (monitoring service): Princeton
  - PlanetFlow (auditing service): Princeton
  - SWORD (discovery service): Berkeley, UCSD

Introduction to PlanetLab

- In this presentation, we focus on the design principles of PlatnetLab
  - “design principles” : the rules that have been recognized and formulated that guide the decisions about how to put the platform together
    - Did not predate the implementation
    - Have co-evolved with the architecture itself

Goals

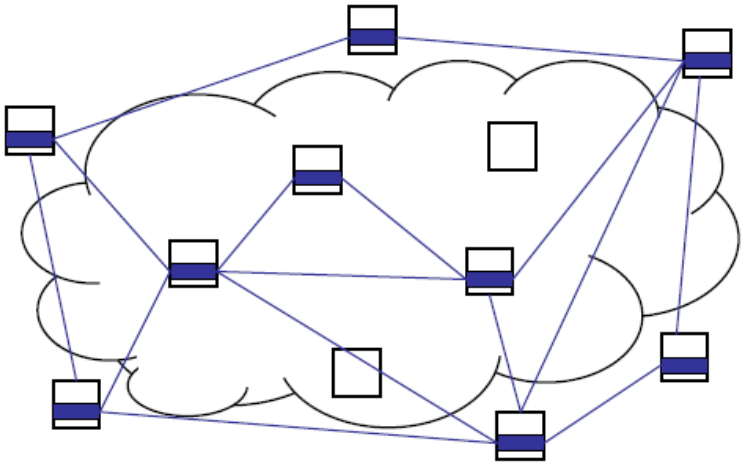
- Underlying the design principles are the high-level goals of PlanetLab
- From the beginning, 3 goals have been identified
  - to provide a platform for researchers to experiment with planetary-scale network services
  - to provide a platform for novel network services to be deployed and serve a real user community
  - to catalyze the evolution of the Internet into a service-oriented architecture.

Virtualization

- Definition
  - a technique for hiding the physical characteristics of computing resources from the way in which other systems, applications, or end users interact with those resources.
  - This includes
    - making a single physical resource (such as a server, OS, application, or storage) appear to function as multiple logical resources
    - making multiple physical resources (such as storage devices or servers) appear as a single logical resource
- Two kinds
  - Platform virtualization involves the simulation of virtual machines.
  - Resource virtualization involves the simulation of combined, fragmented, or simplified resources
    - such as storage volumes, name spaces, and network resources.

Terminology

- A node is a machine capable of hosting one or more virtual machines.
- A virtual machine (VM) is an execution environment in which a slice runs on a particular node
- PlanetLab users who wish to deploy applications acquire a slice,
  - Which is a collection of virtual machines spread around the world
- The VMs are implemented on physical machines by some OS mechanism or virtual machine monitor (VMM), and controlled by the node manager(root VM)
- Today, most policies are hard-coded, but local admin will be able to configure them on their own nodes



VMM: linux kernel(Fedora core)
+ Vservers (name space isolation)
+ Schedulers (performance isolation)
+ VNET (network virtualization)

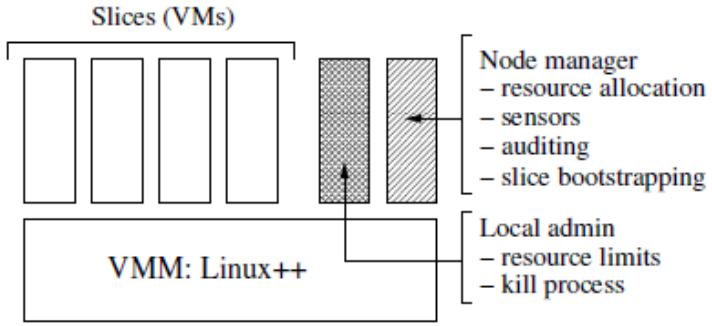
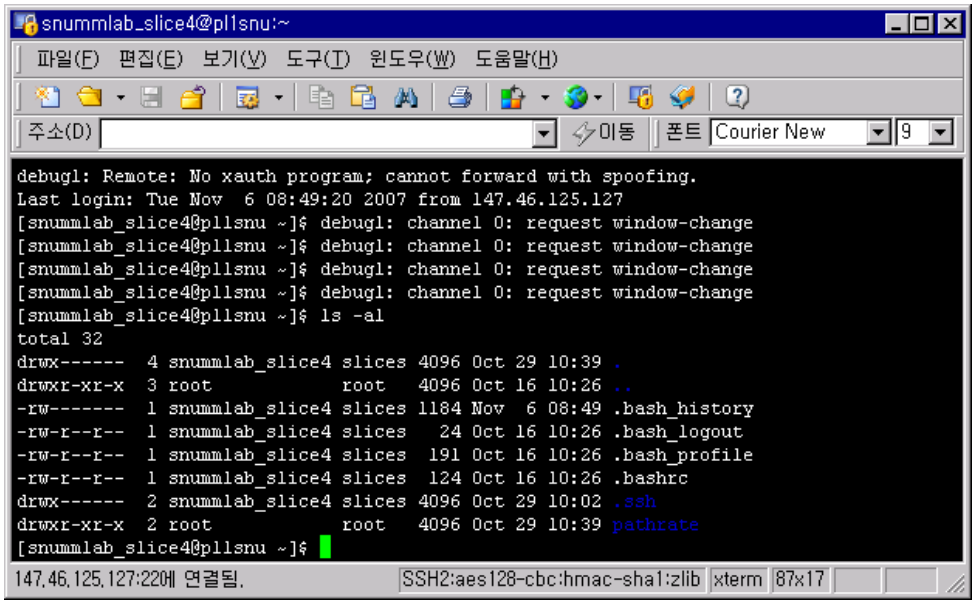


Figure 1: PlanetLab Node Architecture

Terminology

- An infrastructure service
  - a “helper” service used by other slices (services).
  - Ex) create slices on a set of nodes; buy and sell node resources
- The control plane of PL
  - Node managers
  - infrastructure services
  - account management and node installation functions (currently centralized)



Design challenges

- Design challenges
  - Minimize centralized control without violating trust assumptions
  - Balance the need for isolation with the reality of scarce resources
  - Maintain a stable and usable system while continuously evolving it

Distributed virtualization

- What is distributed virtualization
  - The acquisition of a distributed set of VMs that are treated as a single, compound entity by the system
  - For this, PL provides
    - Facility to create a slice, initialize it with sufficient persistent state to boot the service or application
    - Facility to bind the slice to a set of resources on each constituent node
  - Example
    - Allows a slice-specific overlay to be created
    - Provides the means by which slices can install whatever software they need

Unbundled management

- Allow parallel infrastructure services to run in their own slices and evolve over time
- Example
  - Rather than a single privileged application controlling a particular aspect of the OS, the PlanetLab OS potentially supports many such management services
  - Multiple slice creation services

Chain of responsibility

- Chain of responsibility
  - It must be possible to map externally visible activity (e.g., a transmitted packet) to the user responsible for that packet
  - Preserve the implicit trust relationships
- Trust relationship
  - 300 organizations have contributed nodes
  - 375 research groups want to deploy service
  - PlanetLab Consortium (PLC)
    - A trusted intermediary
    - Reduce N*N problem into N+N problem
- Two principles
  - Each interaction b/w PL and the rest of the network must be attributable to a PL user
  - Keep explicit the trust relationships between node owners, authorities, slice users.

Evolution vs. clean slates

- The first version was built quickly from preexisting ideas and techs
- A intuition that much of original designs would have to be reworked as experience increases
- Design principles
  - Avoid “clean slate” designs
  - Design the architecture with an openended view of future evolution
  - Leverage existing software and hardware infrastructure
    - Avoid modifying if they can be used as-is
    - Use readily available software such as OpenSSH, Linux, Unix utilities
  - Implement any new system function at the highest level possible
    - Leads to using least privilege for that function
    - Low levels (VMM or OS) require higher privilege than high levels (a slice with limited privilege)

OS and control plane

- Principles
  - Keep the control plane and the OS orthogonal
    - Don’t pollute the OS interface by adding new functionality, when this can be added to control plane interface
  - Use existing interface semantics as far as possible
    - No operations should be added if the desired function can already accessed through the existing OS interfaces
  - Don’t tackle porting issues
    - The overhead of porting an app should be the same when running on PL or on a native OS

The control plane of PL
- Node managers
- infrastructure services
- account management and node installation functions

VNET

- VNET: PlanetLab Virtualized Network Access
- Connection tracking : the heart of VNET
  - Connections are defined on a per-protocol basis
    - Supported protocols : TCP, UDP, ICMP, GRE and PPTP
  - Relies on Linux's Netfilter system to associate every inbound and outbound IP packet with a connection structure.
  - Connection reservation is done by bind system call
    - Which normally used to specify the interface and port number that a socket should use
    - Once a local port is successfully bound by a slice, no other slice may send or receive packets associated with that port

Conclusion

- Introduction to PlanetLab
  - Basic architecture
  - Terminology
  - Virtualization
- Design issues of PlanetLab
  - Design goals, challenges
  - Design principles
  - Maybe also applied in the future internet in that many heterogeneous systems coexist, in terms of the scale, virtualization, management, etc.