
CS199-6: Wide Area Application Design, Deployment, and Management

<http://cs199.planet-lab.org/>

David Culler
Brent Chun
Timothy Roscoe

Tuesday 20th January 2003



PLANETLAB

Overview of today's talk

- What's this course about?
- Planetary-scale Applications
 - What are they?
 - How are they different?
- PlanetLab as a platform
 - What is it?
 - How does one use it?

What's this course about?

- Writing distributed services to run in the wide area Internet
 - spread over a large no. of machines and large geographical area
 - which can be deployed over PlanetLab
 - which might become part of PlanetLab
- Experience in the design of large systems
 - Network programming
 - Handling failures
 - etc.
- Introduction to real systems research
- Emphasis on *building* rather than lectures

Mothy and Brent's goals

- Give you a feel for what implementing large distributed systems is like
- Help you get experience designing, building, testing, and debugging real wide-area services
- Not to simply feed you information
- => Ask questions, arrange meetings, etc. etc.

Approach

- Programming experience is assumed
- Few introductory lectures
- Reading material
- Get into design and implementation as soon as possible
- Work in teams of 2-3 people

Phase 1: Introduction

- Lectures on
 - Planetary-scale services
 - PlanetLab
- Initial assignment
 - Simple geographical lookup service
- Location: Soda 310

Phase 2: Project Proposals

- Form teams of 3 people for projects
- Put together project proposals
- Review projects
- Location: 310 Soda

Phase 3: Building a service

- Implementation of project proposals
- Weekly meetings with each team
- Milestones:
 1. Initial prototype
 2. Enhancements
 3. Final delivery
- Meeting location: Intel Research
 - (downtown - the PowerBar building)

Phase 4: Reporting

- Each team gives a presentation and demonstration of their service
- Project writeups due
- Location: Soda hall

Tentative Schedule

Week 1:

Introductory lectures

Initial assignment

Week 2:

Project discussion

Form teams

Week 3:

Review proposals

Thereafter:

Implementation and Team meetings

Project Milestones



Milestones

- 28th January: Initial assignment due
- 4th February: Team project proposals due
- 24th February: Initial prototype due
- 17th March: Enhancements to prototype
- 21st April: Deliver final service
- 27th April, 6th May: Presentations/demos
- 16th May: Project writeups due.

Reading material

- See:

<http://cs199.planet-lab.org/reading.html>

- Required reading for next week:

- "A Blueprint for Introducing Disruptive Technology into the Internet"
- "A Note on Distributed Computing"
- "Hints for Computer System Design"



Planetary-scale and wide-area distributed systems

What is a distributed system?

- Distinct components running on distinct machines
 - WWW, NFS, CIFS, Email, Ultima Online, Quake3, Saber, SS7, etc., etc.
- Characterized by:
 - Concurrency
 - Partial failures
 - Latency
- Writing distributed systems is *hard*
 - C.f. Waldo et.al.

Distributed systems: concurrency

- Concurrency can often be dodged in centralized systems
 - Event-driven systems, one-offs
- Alternatively, locks are available
 - E.g. Java concurrency primitives
- Distributed systems are inherently concurrent
 - And shared-memory-based synchronization is not an option

Distributed systems: latency

- Typical procedure call: ~1ms vs. ~10ns.
- High-level system design must take this into account
 - Pipelining
 - Parallelism
 - Etc.

Distributed systems: partial failure

- "A distributed system is in which I can't get my work done because I computer I've never heard of has failed"
 - Butler Lampson
- Dist. Systems are not fail-stop
 - Bits keep running
 - Failures may be undetected
 - Etc.

Distributed Systems

- Despite all this, distributed systems are, these days, relatively commonplace
 - Some are well-engineered
 - » e.g. SS7, Ultima, etc.
 - Some are sufficiently simple
 - » e.g. WWW
 - Some people just live with
 - » e.g. WWW, NFS, CIFS
 - Some people are told to just live with
 - » e.g. most corporate calendaring systems

Wide-Area (or Planetary-Scale) Systems

- Wide area applications are for people who find ordinary distributed applications too easy :-)
- Wide-area applications span a significant portion of the globe
 - *Google is not* a planetary-scale system
 - *Akamai is* a planetary-scale system

Why build planetary-scale systems?

- Latency
 - Beating the speed of light
 - Move computation and data closer to users
- Multilateration
 - Stand in 1000s of viewpoints at the same time
 - Triangulation, correlation, measurement
- Politics
 - Spanning boundaries
 - Selecting (or avoiding) domains
 - judicial, financial, administrative, national, etc

Examples of wide-area systems

- Content-distribution networks
 - Akamai, Inktomi, etc.
- Overlay routing networks
 - RON (Resilient Overlay Networks), etc.
- Global storage systems
 - OceanStore, PAST, etc.
- True Peer-to-peer systems
 - FreeNet, KaZaA, etc.

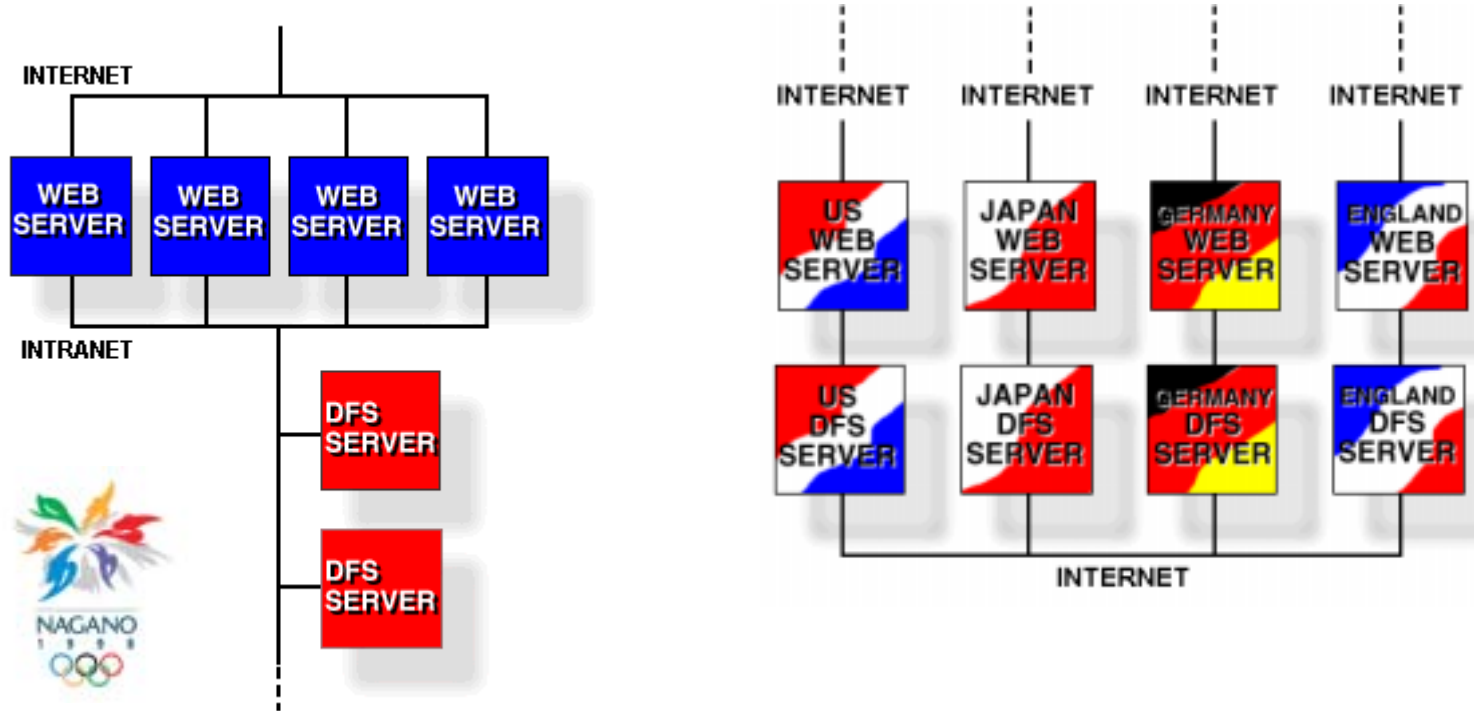
Content Distribution, 1993



- NSCA's "What's New" the most viewed page on the web (100K accesses per month).
- All clients access a single copy of the page stored on a single server.

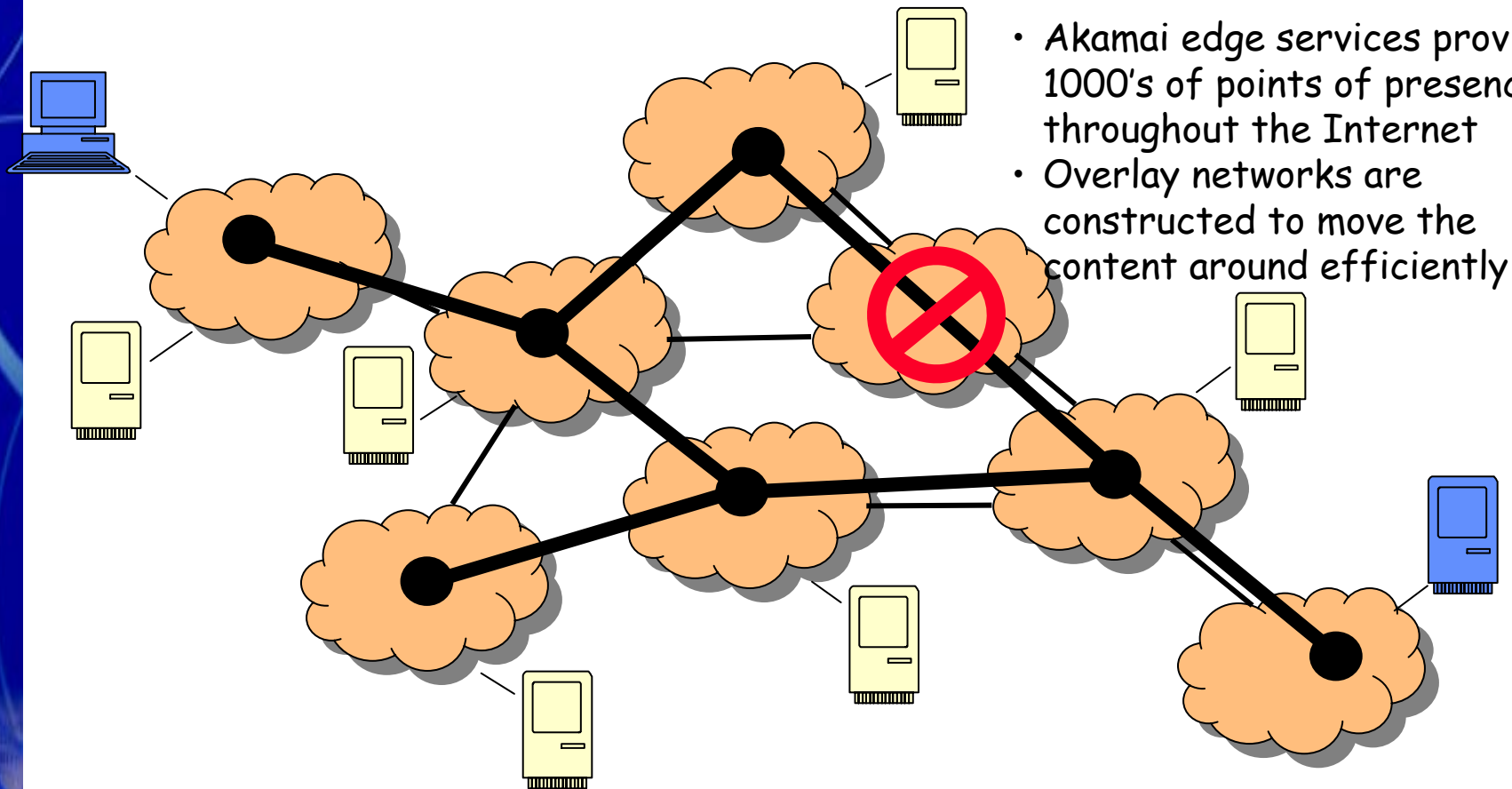


Content Distribution, 1998



- IBM web "server" handles a record 100K hits per minute at the Nagano Olympics
- Over 10^9 pages served in a two week period
- DFS running on SP2's used to distribute 70K pages to 9 geographically distributed locations

Content Distribution Today



- Akamai edge services provide 1000's of points of presence throughout the Internet
- Overlay networks are constructed to move the content around efficiently

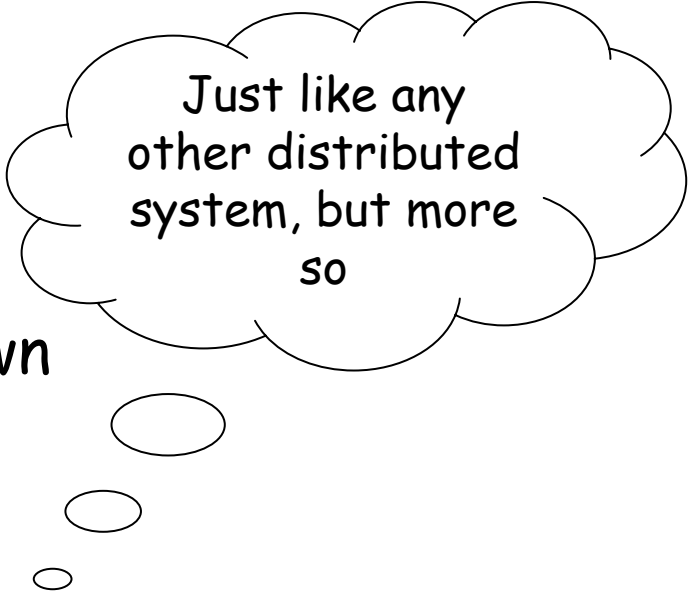
To be effective, the application level overlay network must adapt to changes in the underlying Internet

Other examples:

- See the reading list
- Of perhaps particular interest:
 - FreeNet
 - » & Mnemosyne, etc.
 - Chord
 - » & Tapestry, Pastry, CAN, etc
 - Oceanstore
 - » & CFS, PAST, etc.
 - Resilient Overlay Networks
 - » & End System Multicast, etc.

What's hard about these systems?

- Scalability
 - > 100,000s users
- Reliability
 - System should never go down
- Performance
 - It shouldn't suck
- Management
 - How does something this big stay manageable?



Just like any other distributed system, but more so

What's hard (and new) about these?

- Heterogeneity

- Lots of different machines, and different components which have to talk to each other

- Security

- We're now spanning organizational boundaries
- Perimeter-based security doesn't really work

- Evolution

- Parts of the system must change incrementally over time
- We can't just restart everything.

Wide-area application research

- Lots of recent research work:
 - RON, ESM...
 - Storage: Oceanstore, IBP, CFS, Past...
 - DHTs: Tapestry, Chord, CAN, Pastry...
 - Event systems: Scribe, Herald, Bayeux...
 - CDNs
- Results tend to be based on:
 - Simulation
 - Emulation (clusters, etc.)
 - Small-scale deployment (call your friends)



PlanetLab: What and Why?

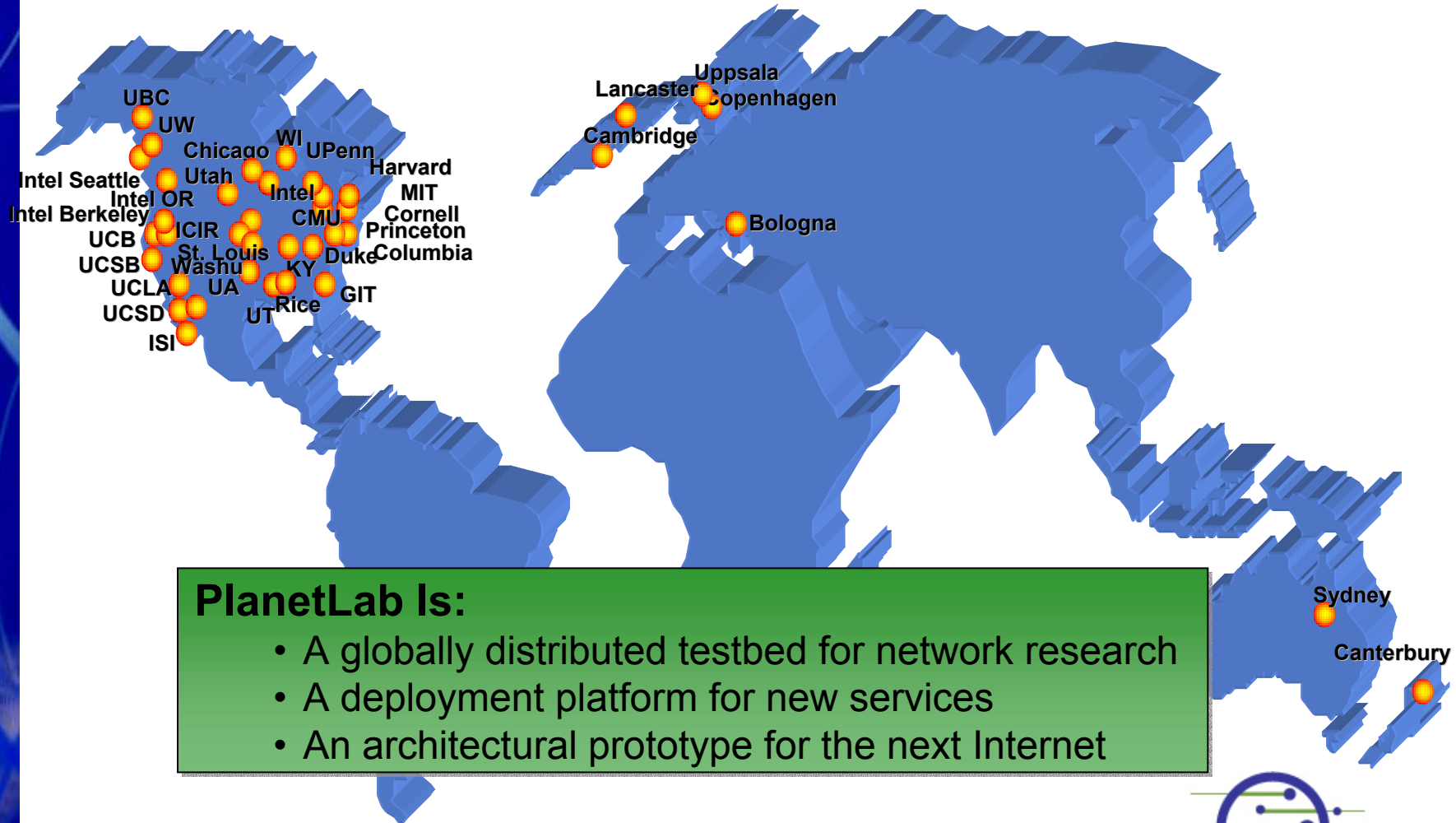
Doing all this for real...

- ... is hard for a researcher
- Where do you get access to 1000 geographically dispersed machines?
- How do you do it legitimately?
 - No worms
 - No cracking
 - No Venture Capital

So what is PlanetLab?

- An open, shared testbed for developing, deploying, and accessing planetary scale services
- <http://www.planet-lab.org>
- Boils down to:
 - A set of machines to run your code on all over the world
 - An operating system to make this safe
 - Management software to keep it working
 - Useful services to save you time and effort

PlanetLab



PlanetLab Is:

- A globally distributed testbed for network research
- A deployment platform for new services
- An architectural prototype for the next Internet

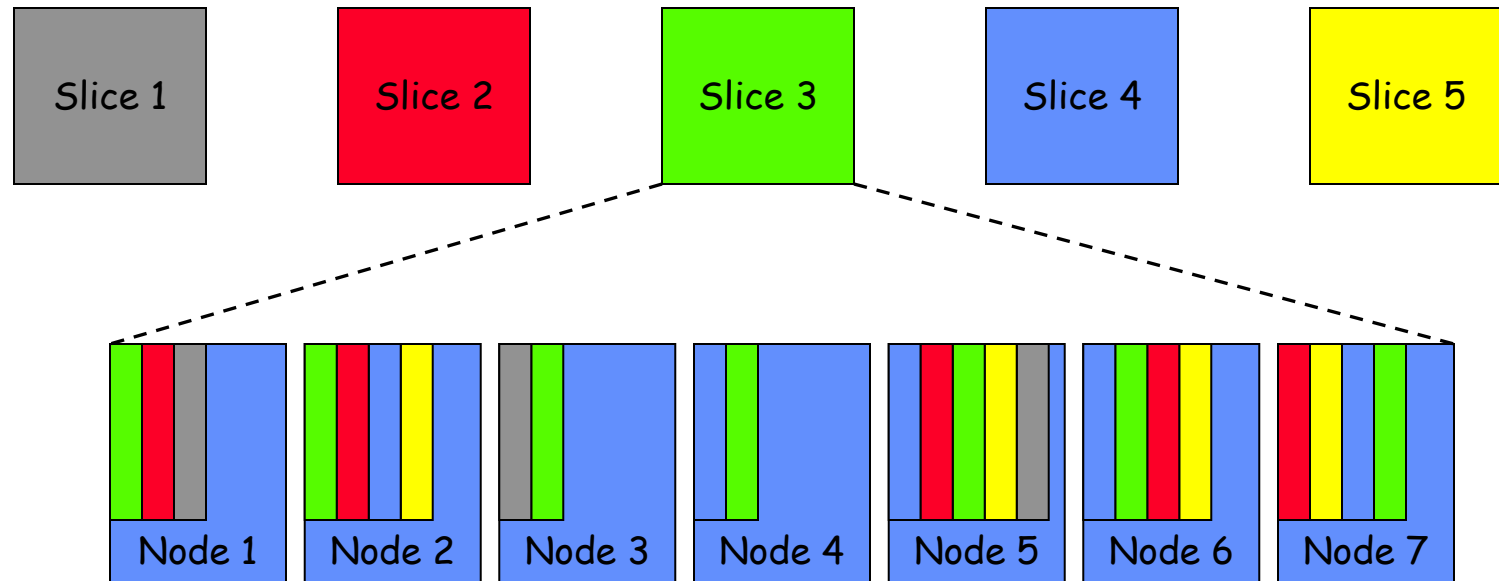
PlanetLab

- A testbed for network research
 - 1000 nodes at various sites around the Internet
 - » Geographically distributed
 - » Distributed across networks
 - A representative sample of the Internet
 - » Diverse bandwidth & latency
- A deployment platform for new services
 - Apps: content distribution, distributed DOS response
 - Middleware: overlay networks, DHT's
 - Network: traffic measurement
- The next Internet
 - The network is designed for extensibility
 - Extend the network upwards, push applications down

Distinguishing Properties

- **Slice-ability**
 - Each service runs in a slice of PlanetLab
 - VMM on each node enforces slices
 - Global admission control policy
- **Distributed control of resources**
 - Services deployed over a chosen set of nodes
 - Local control of services that run on a node
- **Unbundled management**
 - Partition management into orthogonal services
 - Core services versus competing alternatives
- **Application-centric interfaces**
 - Stable platform versus research into platforms
 - Separate isolation interface and application interface

PlanetLab Slices/Slivers



A network service is broken into components that can be distributed throughout the Internet

- Slice: total resources for the service
- Sliver: resources required on a specific node

What does PlanetLab give you?

- PlanetLab gives you (yes you!)
 - Ability to deploy a service around the world
 - Chance to contribute to the research
- PlanetLab is under development
 - Management services
 - Naming and location
 - Etc.
- PlanetLab is shared
 - Responsible use is called for!

Tomorrow:

- More detail on:
 - how to use PlanetLab
 - That initial project assignment
 - How to contribute project proposals
 - Some we've thought of

- Meantime, questions?