GENICLOUD: AN ARCHITECTURE FOR THE INTERCLOUD

Alvin Au Young, Andy Bavier, Daniel Catrein, Jim Chen, Jessica Blaine, James Kempf, Christian Lottermann, Joe Mambretti, Rick McGeer, Alex Snoeren, Marco Yuen

UCSD, iCAIR, PlanetWorks, HP Labs, University of Victoria

July 23, 2010

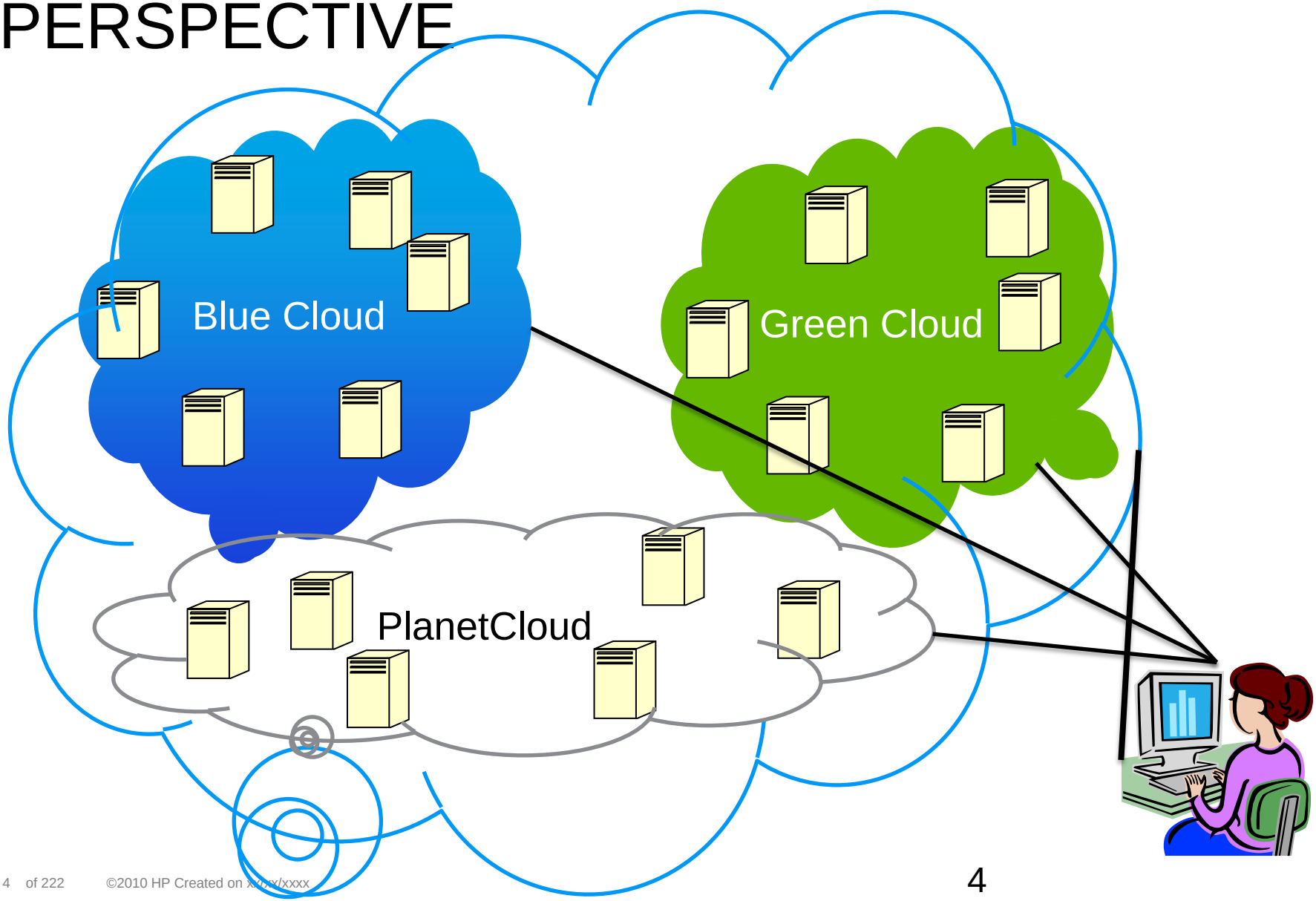
WHY PLANETLAB AND THE CLOUD?

- Strengths of (existing) PlanetLabFacilities
  - Broad global reach
  - Large (aggregate) bandwidth and low latency to everywhere
  - Point of presence everywhere on earth
- Weaknesses of PlanetLab facilities
  - Not much computation available anywhere
- Strengths of the Cloud
  - Large chunks of computation available
- Weaknesses of the cloud
  - Bandwidth limited to a few centers
  - Latency variable

MOTIVATION: THE INTERCLOUD

- Internet: set of standards and protocols which permit interconnection of independently-administered networks
  - Network of networks
- Intercloud: Set of standards and protocols which permit interconnection of independently administered clouds
  - Term due to Greg Papadopoulos
  - Defining infrastructure of 2010's and beyond
- Question: What will the Intercloud look like? What makes it an intercloud (as opposed to a Cloud)?

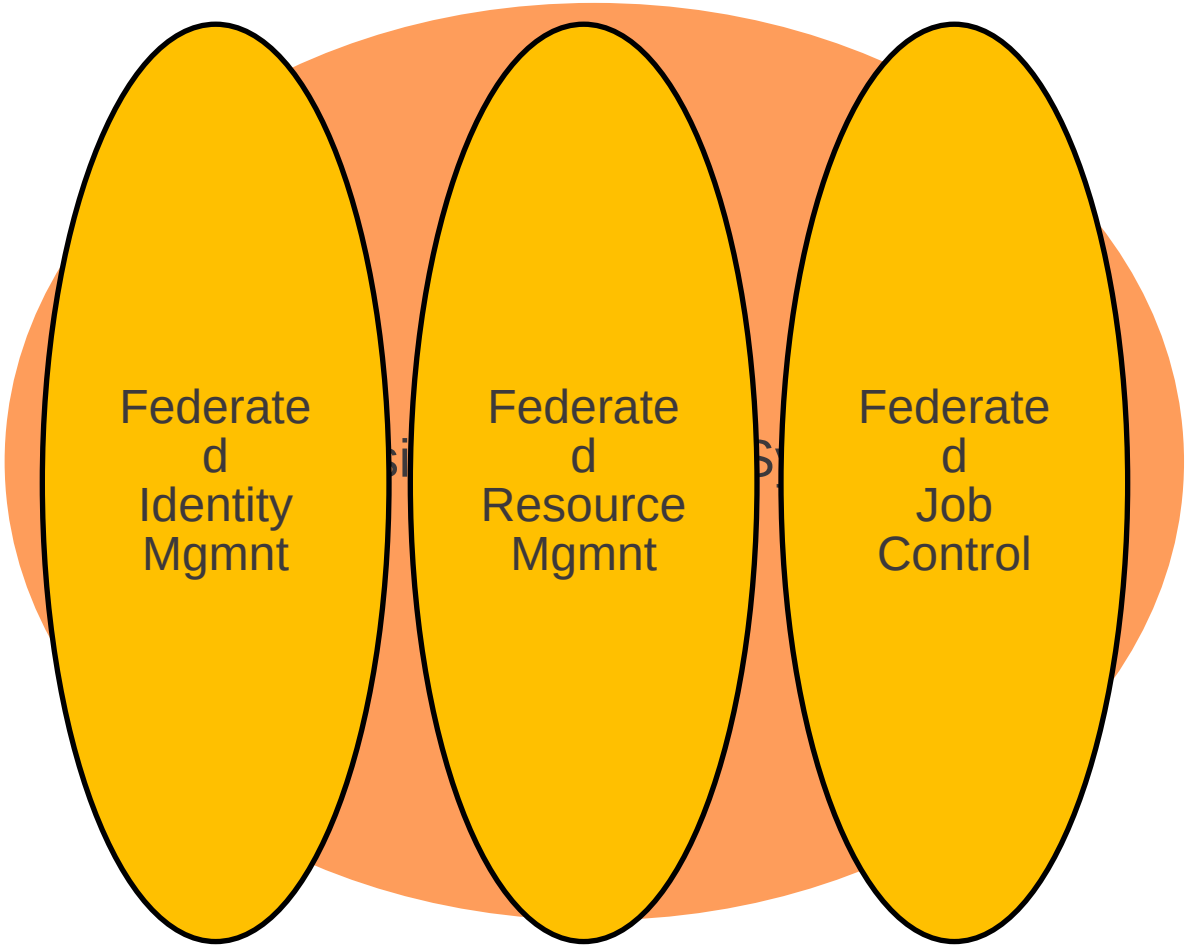
INTERCLOUD FROM A DEVELOPER'S PERSPECTIVE



REALIZING THE INTERCLOUD FROM A DEVELOPER'S PERSPECTIVE

- Key question: what are the *minimal* set of constructs/principles required to permit this interconnection?
- This really amounts to a set of agreements between Blue Cloud, Green Cloud, PlanetCloud
- Note that *any* agreement is *very* hard
  - BlueCloud, GreenCloud, PlanetCloud operate in different environments, different constraints
  - Even absent "walled garden" incentives, getting deep agreement is very hard
  - Ex: US law treats clusters as "munitions", restricts access to persons of various nationalities (folklore, haven't been able to nail this down)
- So…what do we really need?
- Testbed: OpenCirrus, OpenCloud, and PlanetLab
  - Three+ separate domains, two different slice managers,

FEDERATION



KEY ASSUMPTION

- Each facility implements Slice-Based Facility Interface
- GENI standard for control frameworks
- Standard, unified means of allocating
  - Virtual machines at each layer of the stack (“slivers”)
  - Networks/sets of virtual machines (“slices”)
- Already supported by PlanetLab, ORCA
- Now supported by Eucalyptus (our contribution)
- Should be easy to port to Nimbus, etc

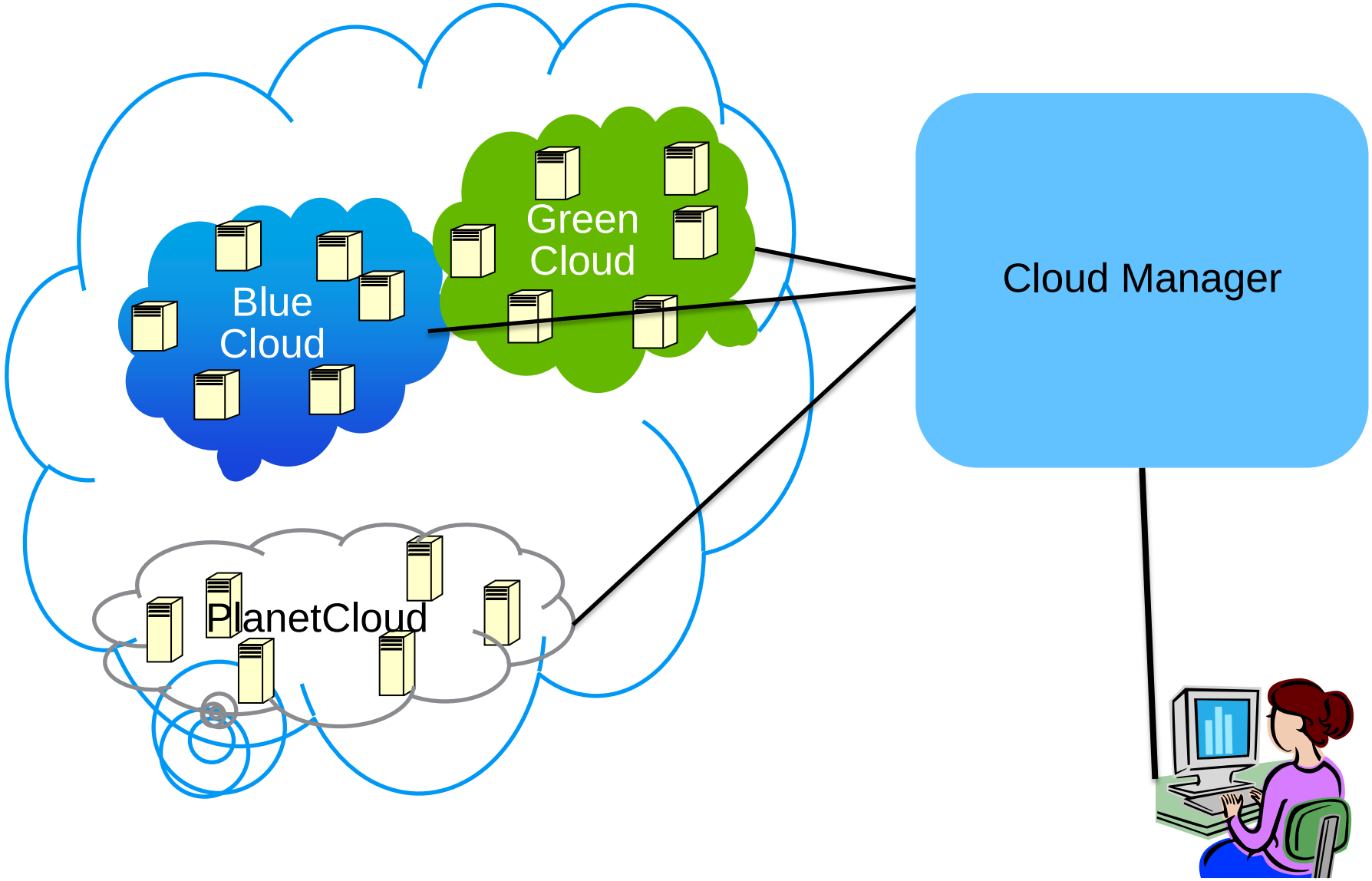
WHAT WE NEED, WHAT WE DON'T

- What we need
  - Method of creating slices on clouds and distributed infrastructures
  - Method of communicating between clouds and distributed infrastructures
  - Method of interslice communication between clouds
- What we don't
  - Single sign-on!
  - Single AUP
  - Single resource allocation policy or procedure
  - Unified security policy
- Principle of Minimal *Agreement*
  - What is the minimum set of standards we can agree on to make this happen?

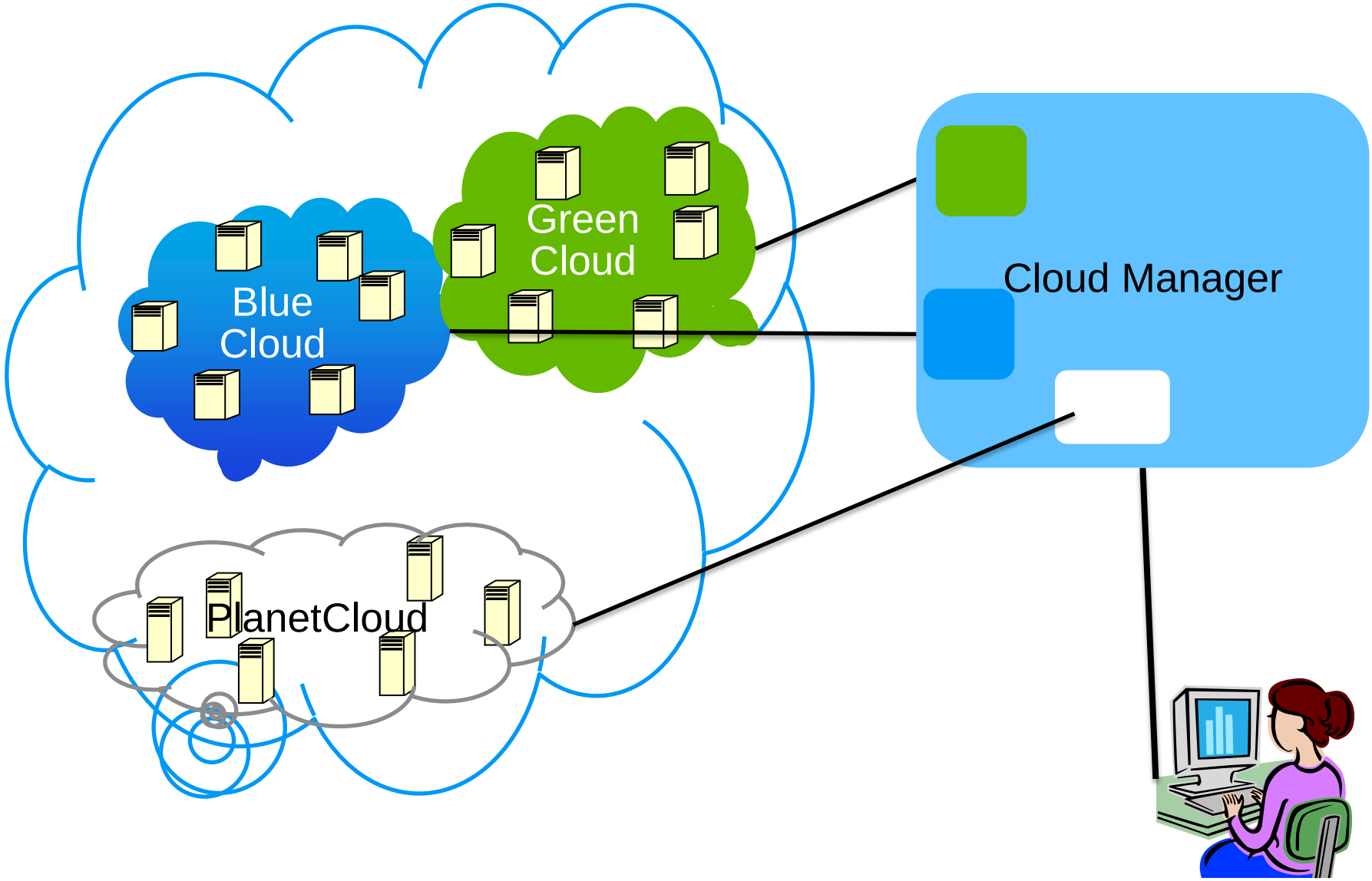
FIRST LAW OF NETWORKING

- Every problem can be solved with one more proxy
- In this case: need a *Cloud Manager*
  - User-appointed delegate to make the Intercloud look like a single Cloud
- Key:
  - Each individual cloud oblivious to the existence of the other clouds
- Analogy to proxies
  - Proxies bridge discontinuities on the Internet
  - Cloud Managers bridge discontinuities in the Intercloud (naming, user IDs, AUPs, resource allocation…)

ROLE OF THE CLOUD MANAGER



CLOUD MANAGER: PLUG-IN ADAPTERS



GENICloud Architecture

Aggregate Manager (PLC)

Slice Manager

User-facing Cloud Proxy With Distributed Rate Limits

Node Manager Sliver

Aggregate Manager

Front End

Node

Node

Node

Eucalyptus Cluster 1

Aggregate Manager

Front End

Node

Node

Node

Eucalyptus Cluster 2

WHAT DO WE NEED FROM THE CLOUDS

- Building Blocks
  - Eucalyptus: Open-source clone of EC-2
  - Widespread developer mindshare (easy to use, familiar)
- What we want: Slice Facility Architecture
  - Means of creating/allocating slices
  - Authorization by ssh key (GID)
  - Delegation primitive
  - Explicit costs/resource allocation primitives
    - Need to be able to control costs for the developer

CONCRETE ACTION

- *Every federate must support SFA API!*
  - Brought up SFA under Eucalyptus (Marco and Andy)
- Still TBD
  - Interconnect networking primitives
    - Certainly need DNS-level support
    - What else do we need?

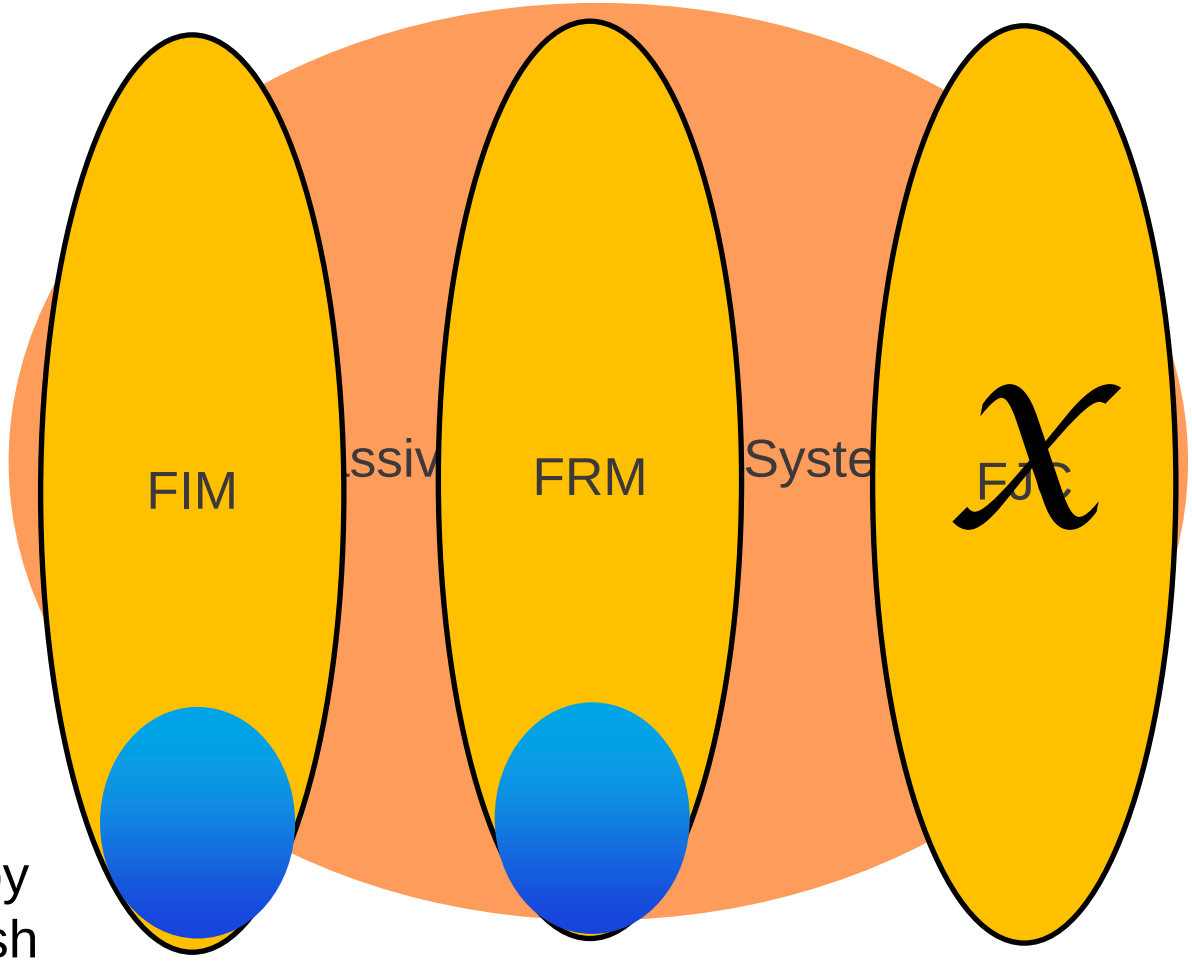
PERMISSION AND DELEGATION

- Basic protocol
  - User creates account on individual clouds
  - Appoints Cloud manager as his delegate
  - Delegates have restricted privileges
    - Create, instantiate slices
    - Upload images to slices
    - Execute jobs on slices
    - Cannot subdelegate
- Permissions
  - Done by key pair
  - Goal: User doesn’t know delegate’s private key
  - Delegate doesn’t know user’s private key
  - Delegate’s key used for no other purpose (easy revocation)

RESOURCE ALLOCATION

- At a high level, simply a uniform specification for resources and prices
  - RSPEC should work just fine
  - Need to add storage primitive (Elliot Jaffe)
- No global policy mechanisms
  - Up to me to make my own deal with each cloud
  - Cloud Manager simply spends for me to get my stuff done
- Cooperation between Green and Blue and Planet…
  - Fine, but Not My Problem.
- Acquiring resources…
  - This stuff usually works…

FEDERATION



Unified *form* of identity (by choice, ssh key)

RSPEC

IS THIS THE END OF THE ROAD?

- Of course not
- But Blue/Green/Planet Federation at a facility level will be a *lot* easier after they see what users actually do…
- History of BGP
  - EGP version 1 (RFC 827, 1982)
  - Full specification (RFC 904, 1984)
  - Use on the NSFNet Backbone (RFC 1092, 1093, 1989)
  - BGP Experimental Specification (RFC 1105, 1989)
  - Proposed standard (RFC 1163, 1990)
  - BGP-3 (RFC 1267, 1991)
  - BGP-4 (RFC 1771, 1995)
  - 20 drafts later – current BGP, BGP4, (RFC 4271, 2006)
- We’re closer to RFC 827 than RFC 4271…

NETWORK STITCHING

- Question open for debate: what should we do here?
- Tussle: deep networking capabilities vs demands on the infrastructures.
- Options:
  - None at all (PlanetLab)
  - DNS Level (no use case for layer 2 or 3) (naming)
  - Layer 3 VLANs
  - VLANs modulated by OpenFlow switches

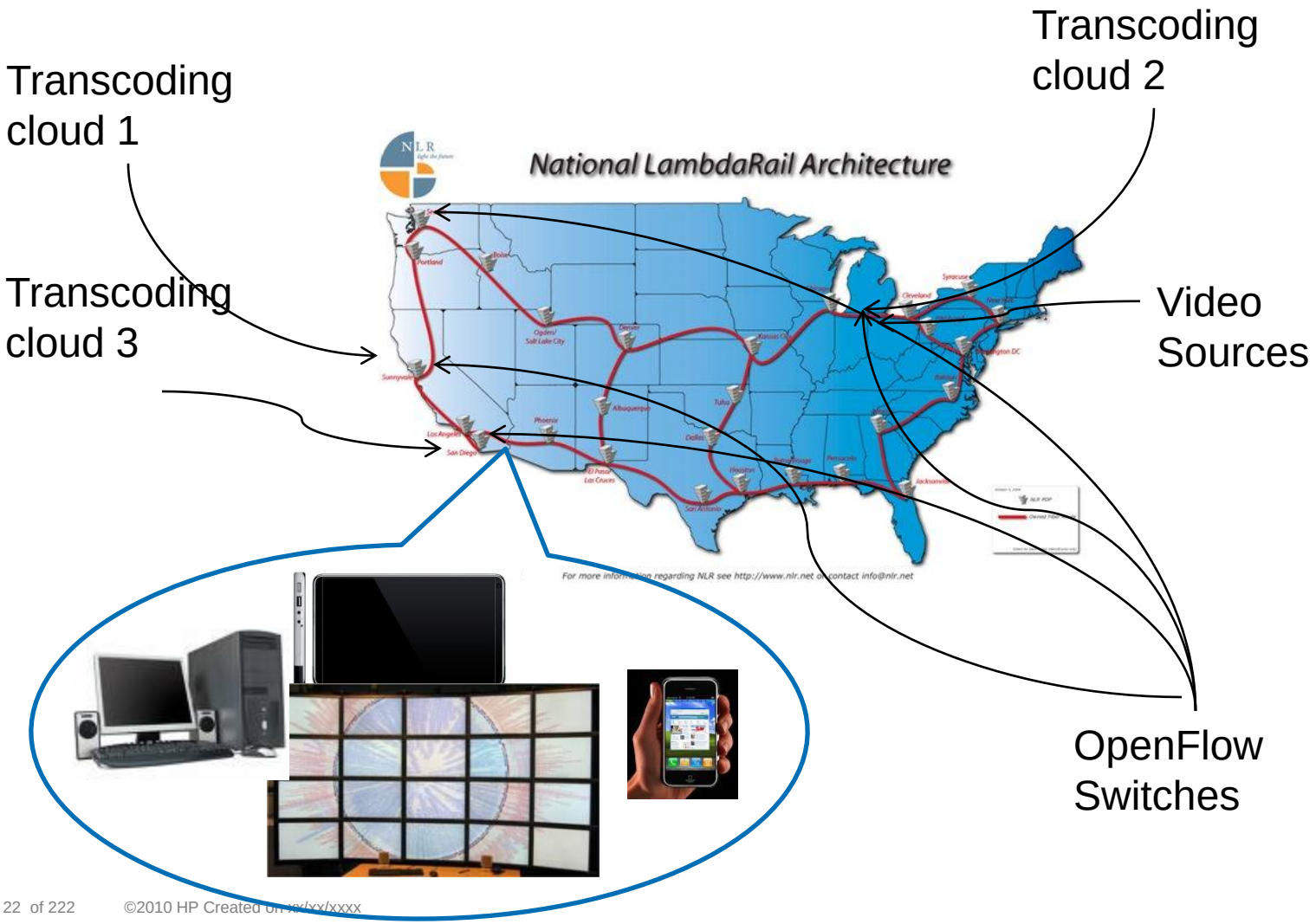
ARE CLOUD MANAGERS GOOD FOR EVERYTHING?

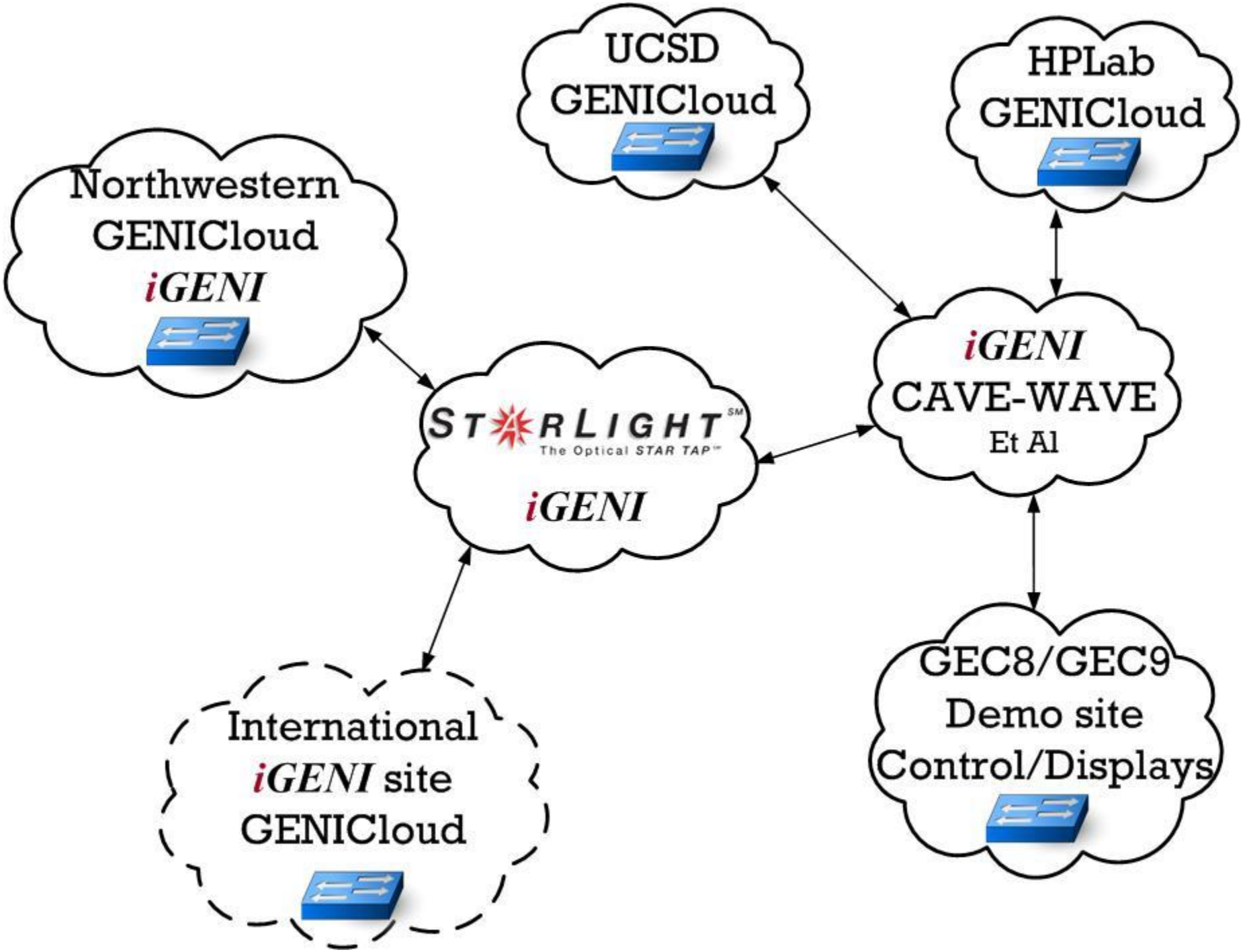
- NO! Need some minimal size of Cloud
  - Else transactions costs kill you
  - User needs to make individual deal with Cloud provider, hand delegation to Cloud Manager
  - Could never have worked for PlanetLab…

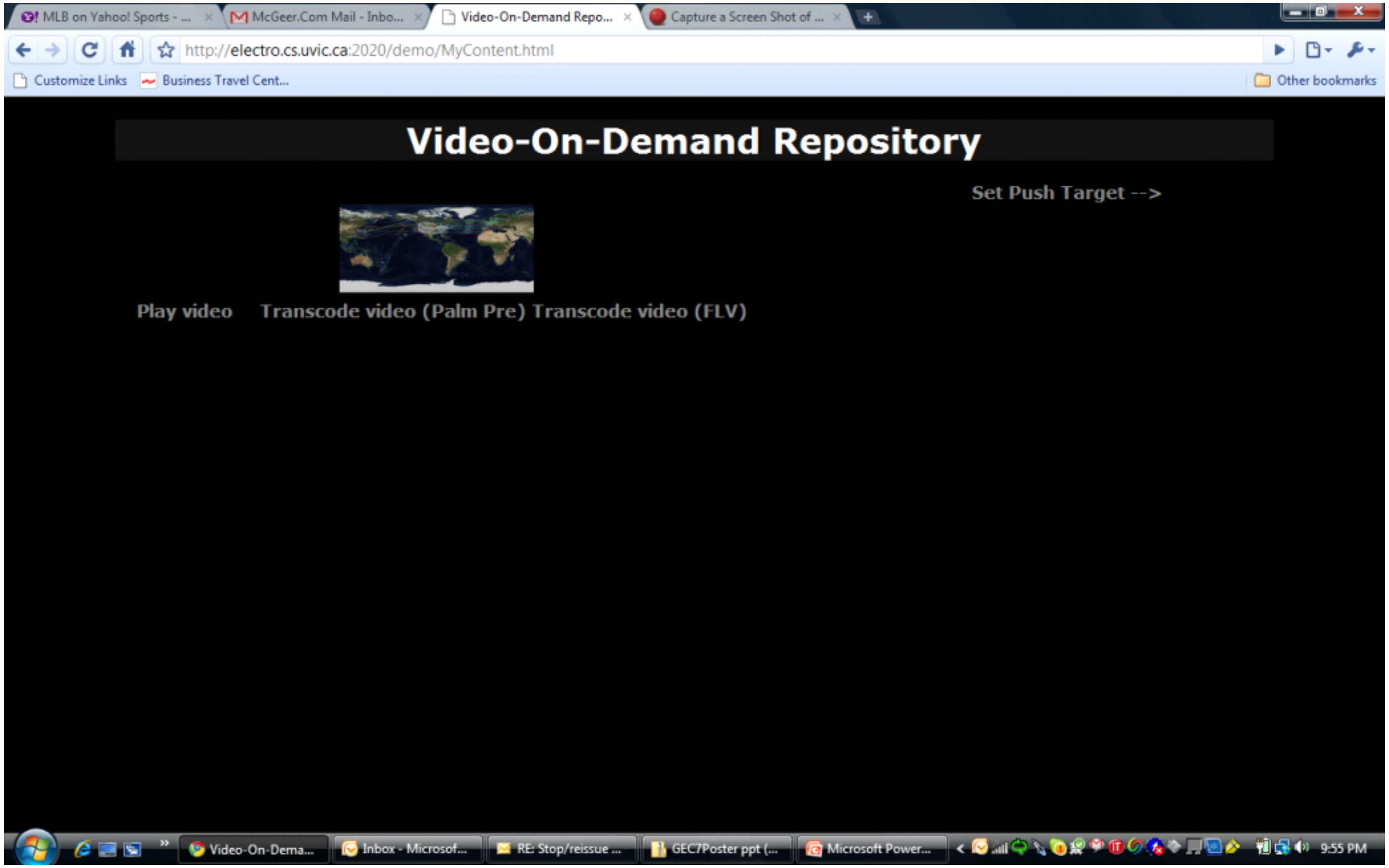
DEMO/PROOF POINT

- Build Cloud Manager and interconnect OpenCloud, OpenCirrus, PlanetLab
- First step: Eucalyptus supporting Slice-Based Facility Architecture
- Second step: Use it to support federated application
  - GEC-8
  - Transcoding cloud service running at UCSD, Northwestern, OpenCirrus
  - Service provided by Ericsson (thanks to James Kempf)

GENERAL ARCHITECTURAL PICTURE




UCSD
GENICloud
HPLab
GENICloud
Northwestern
GENICloud
iGENI
iGENI
CAVE-WAVE
Et Al
STARLIGHT
The Optical STAR TAP
iGENI
International
iGENI site
GENICloud
GEC8/GEC9
Demo site
Control/Displays


MLB on Yahoo! Sports - ...
McGeer.Com Mail - Inbo...
Video-On-Demand Repo...
Capture a Screen Shot of ...
http://electro.cs.uvic.ca:2020/demo/MyContent.html
Customize Links
Business Travel Cent...
Other bookmarks
Video-On-Demand Repository
Set Push Target -->
Play video
Transcode video (Palm Pre)
Transcode video (FLV)
Video-On-Dema...
Inbox - Microsof...
RE: Stop/reissue ...
GEC7Poster ppt (...
Microsoft Power...
9:55 PM

THANKS!

WHY GENI AND PLANETLAB?

- Strengths of PlanetLab
  - Broad global reach
  - Large (aggregate) bandwidth and low latency to everywhere
  - Point of presence everywhere on earth
- Weaknesses of PlanetLab
  - Not much computation available anywhere
- Strengths of the Cloud
  - Large chunks of computation available
- Weaknesses of the cloud
  - Bandwidth limited to a few centers
  - Latency variable

APPLICATIONS OF GENI X PLANETLAB

- General paradigm:
  - Use PlanetLab to send data to/from one or more Cloud Data Centers
    - BitTorrent-like CDNs….
  - Use PlanetLab to *collect* data from real-world testbeds
  - Use Cloud centers to do heavy computation
  - Use Cloud centers for persistent storage
- Second note:
  - Cloud Federation adds scalability, flexibility
  - GENI is inherently federated
  - Use GENI Federation Architecture to offer federation options for clouds

SOME EXAMPLE APPLICATIONS

- Office in the Cloud: use PlanetLab nodes as a cache for personal docs, email, checkpointing
  - Reduce bandwidth need at cloud center
- Local collection, local reduction, analysis in the Cloud
  - Measurement plane on PlanetLab
  - Wide-area distributed experiments (e.g., Cooperative Atmospheric Sensing Experiment)
- Data distribution to Cloud centers for analysis
  - Virtual astronomy, physics mining
- Peer-to-peer Twitter
  - Use social network to overcome locality-of-reference problem in Twitter

SOME EXAMPLE APPLICATIONS

– Data preparation in the Cloud, Data Distribution via GENI
  • E.g., Transcoding media for device form factor, distribution via CDN