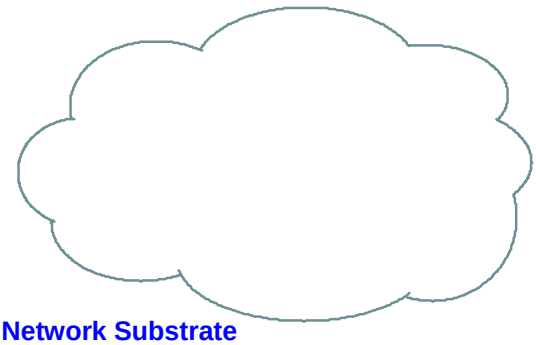
Towards the Big "C" Cloud

Larry Peterson
Princeton University

Scaling Services



Applications

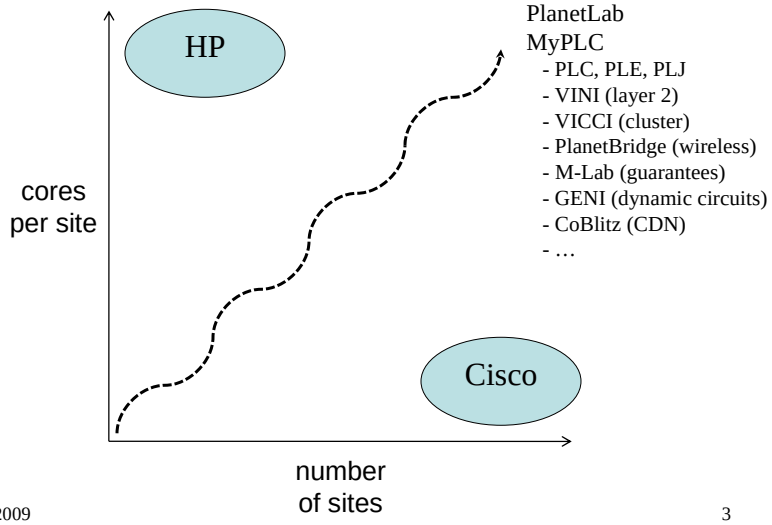
Service innovation starts at the edges

Network Substrate

Substrate should lower the barrier-to-entry.

Scaling successful services requires migrating into the infrastructure

Design Space



PlanetLab
MyPLC
- PLC, PLE, PLJ
- VINI (layer 2)
- VICCI (cluster)
- PlanetBridge (wireless)
- M-Lab (guarantees)
- GENI (dynamic circuits)
- CoBlitz (CDN)
- …

Design Space

- Dimensions
  - 1 – 1,000 sites
  - 1 – 1,000 cores per site
  - 1 – 1,000 autonomous owners
  - 1 – 1,000 applications/services

Design Space

- Differentiating Properties
  - elastic/scalable – pay as you go
    - does not imply “as many free resources as I can get”
    - resource need is workload driven
  - network brings some value
    - not just as happy if all resources in your machine room
    - reduced latency + scalable throughput
    - availability (independent failures)
  - managed and dedicated
    - not individual (p2p) or institution (Grid) contributed

Design Space

- Applications / Programming Models
  - information plane – ping/traceroute
  - routing – send/receive
  - content distribution – get/post
  - distributed file system – read/write
  - distributed database – select/join
  - distributed processing – map/reduce

Common Problems

- Federation – multiple owners
  - common set of abstractions
    - slice – global account / network-wide resource container
    - component – encapsulates a set of resources (e.g., a node)
    - slice authority – vouches for a set of slices
    - management authority – sets usage policy for a set of nodes
  - common set of interfaces
    - principals – Register, Lookup, Update, Delete, GetCredential
    - slices – CreateSlice, DeleteSlice, StopSlice, StartSlice
  - common security architecture
    - all operations require credentials
    - credentials explicitly delegated

Common Problems

- Virtualization – multiple services
  - hyperviser-based vs. container-based
- Remote Management
  - monitoring is a good start
    - automated anomaly detection is necessary for scale
  - ability to diagnose production systems a must
    - profiling detail at monitoring overheads
  - other stuff you need
    - iLO support, ability to bring into safe state, traffic auditing
  - application helper services
    - package management, resource discovery, data logging

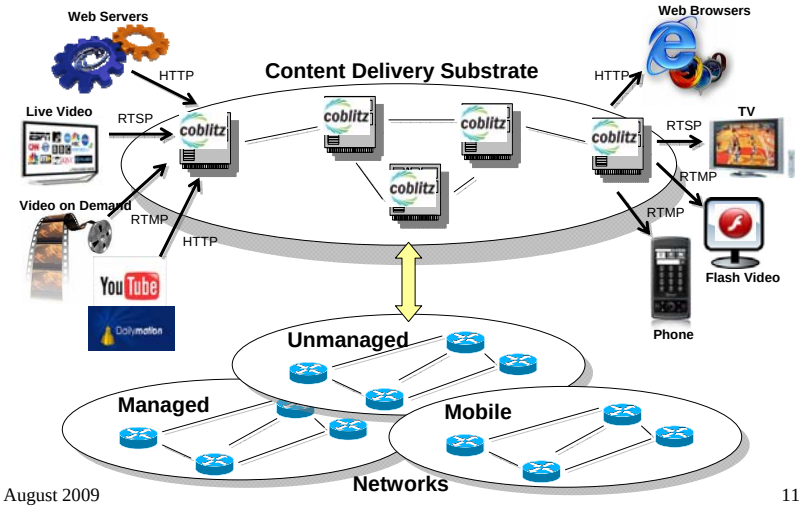
Common Problems

- Resource Allocation
  - support both best-effort and guarantees
    - ↗ don't under value best-effort + controlled-load
  - decouple slice creation and resource allocation
  - separate policy/markets from scheduling mechanisms
- Heterogeneity
  - node: treat as an optimization, not a requirement
  - network: IP is a perfectly good data plane

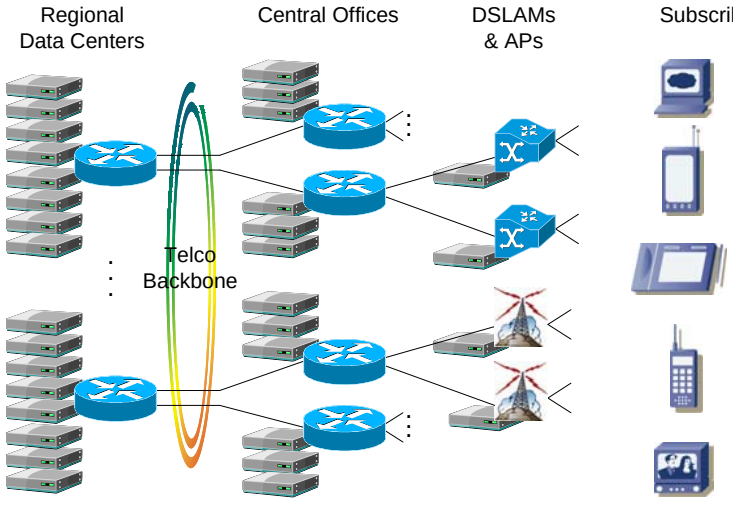
Pull to the Edge

- One potential path forward…

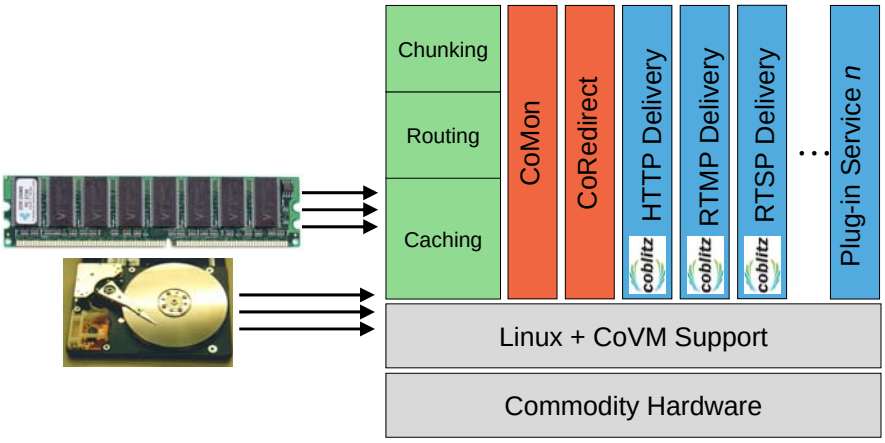
Convergence



Scalability



Extensibility



Conclusions

- Interoperability… *defining a narrow waist*
  - slice – federated container for service deployment
- Cloud >> Data Centers
  - bandwidth scales at the edge of the network
- CDNs (content) provides an "application pull"
  - compelling reason to put cycles/storage in the net
- Natural home for various functions
  - compute-heavy tasks in the core
  - data/state caching at the edge