PlanetLab: Evolution vs Intelligent Design in Global Network Infrastructure

Larry Peterson
Princeton University

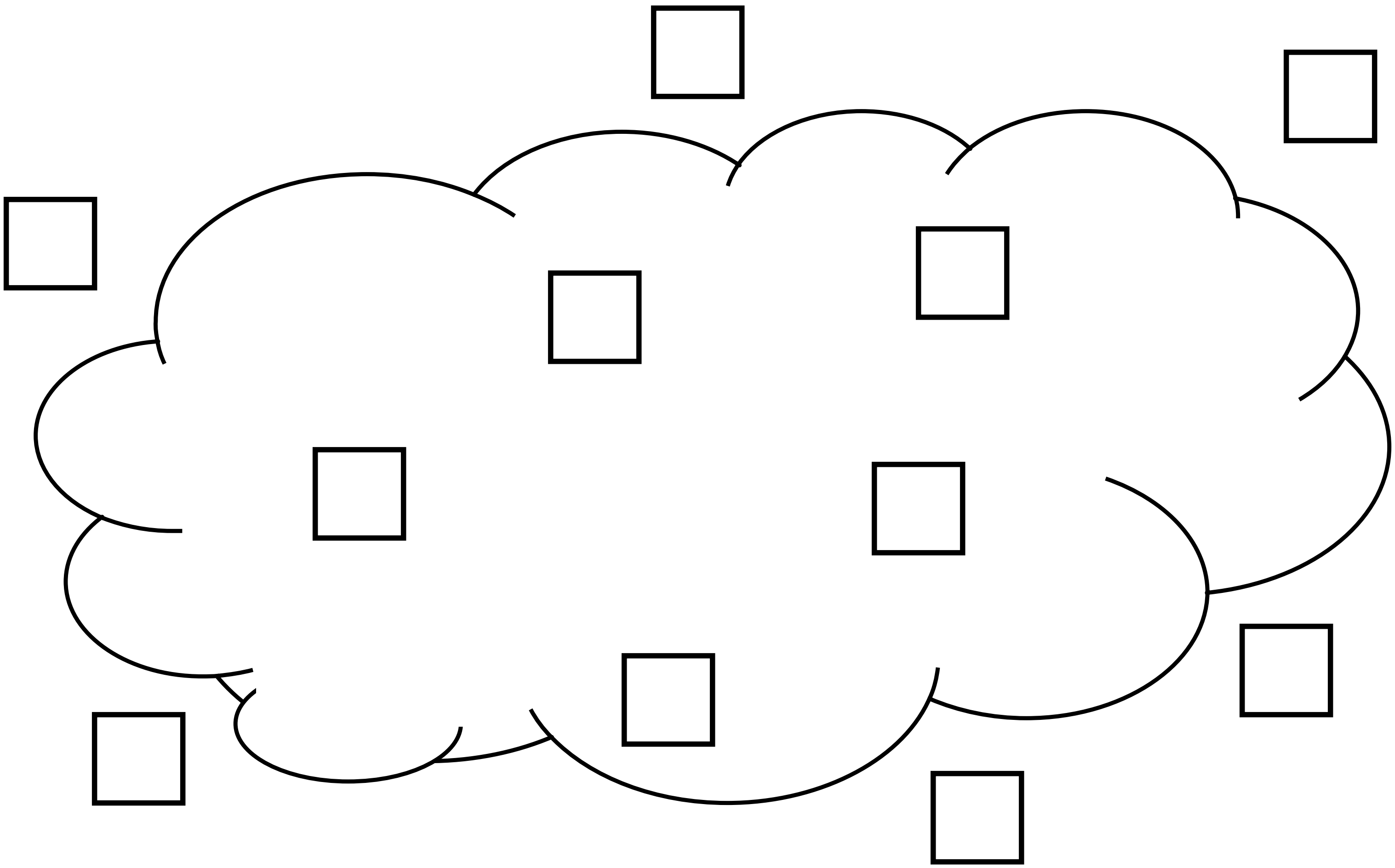
PlanetLab

- 670 machines spanning 325 sites and 35 countries

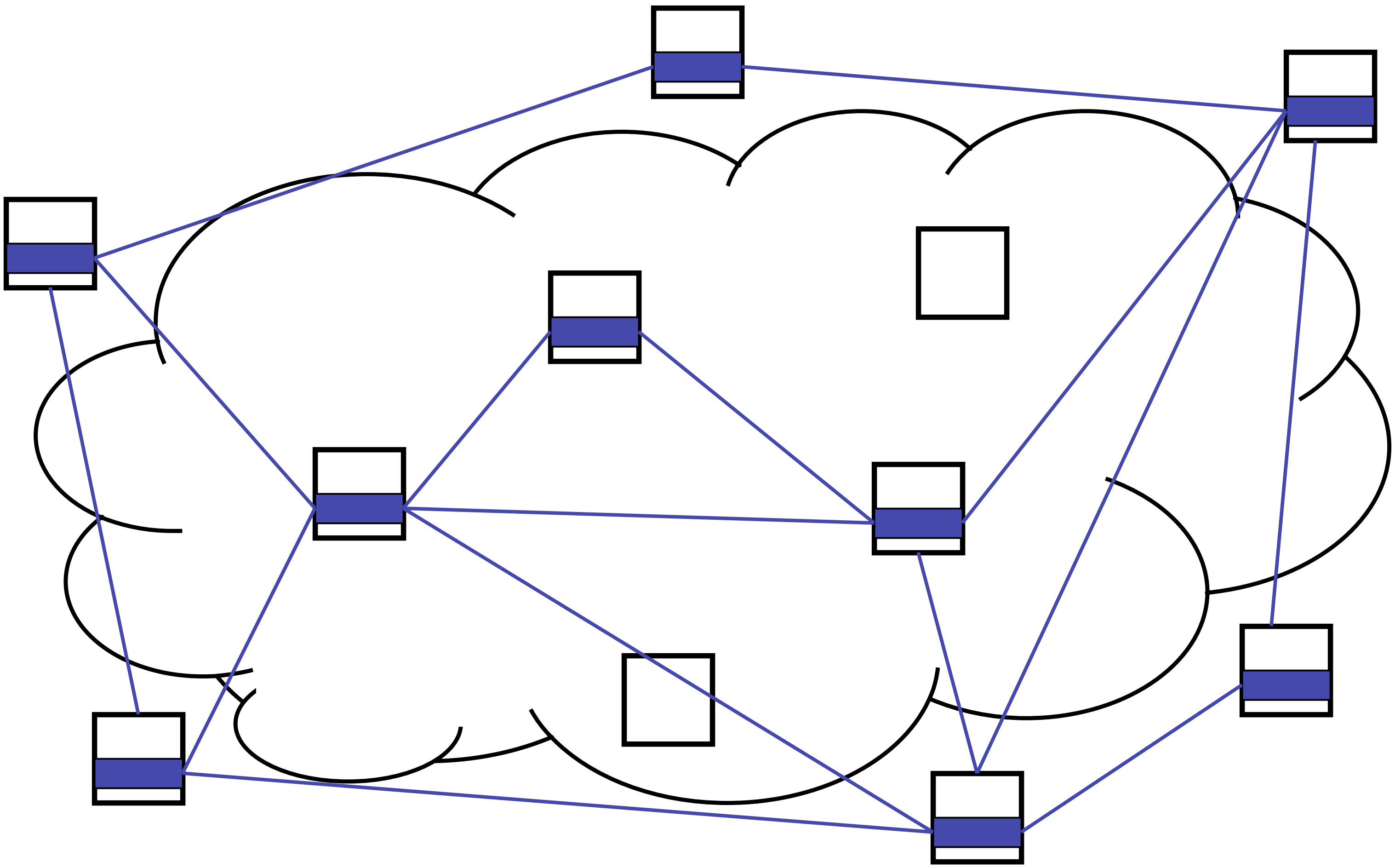
  nodes within a LAN-hop of > 3M users
- Supports *distributed virtualization*

  each of 600+ network services running in their own *slice*

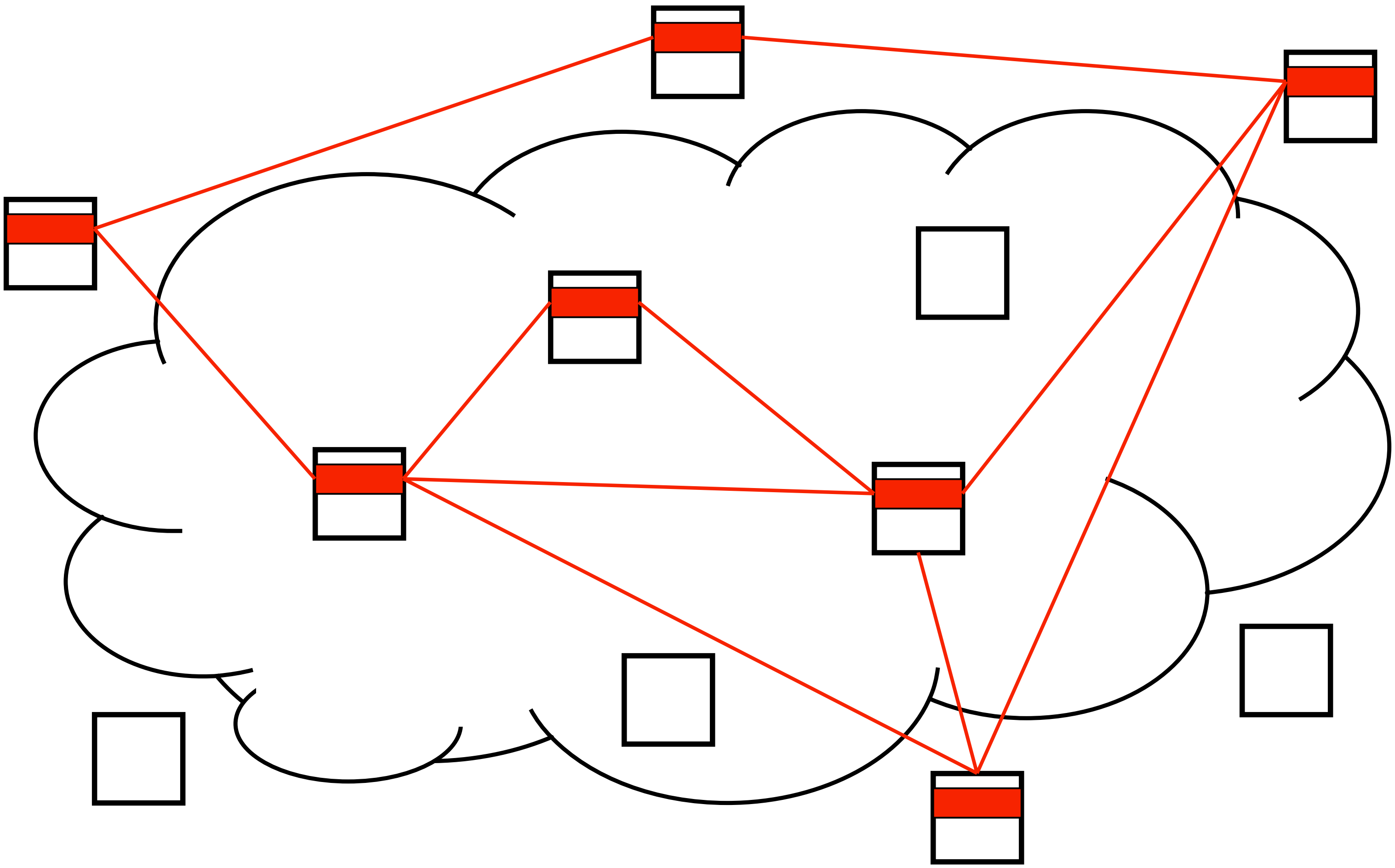
Slices



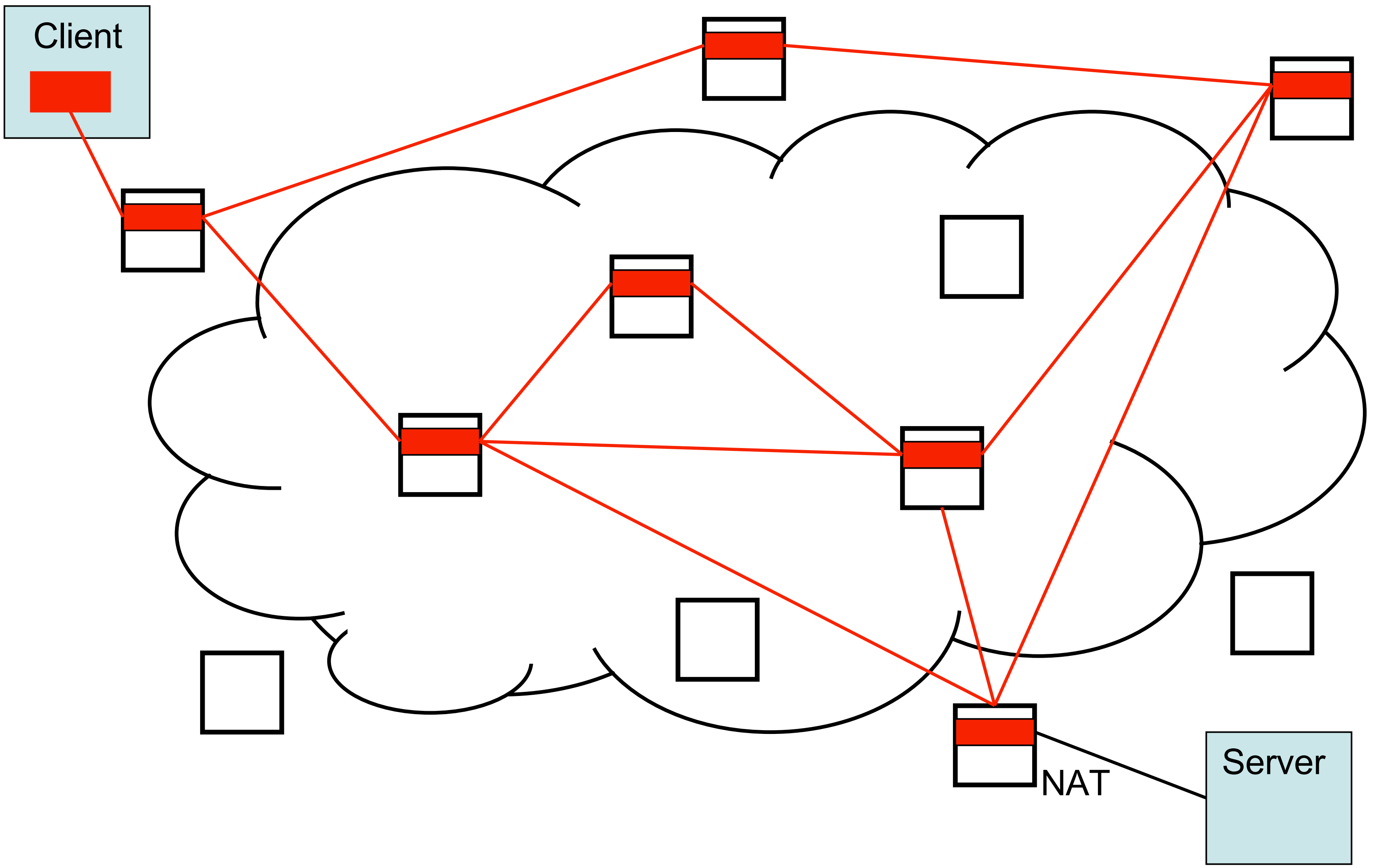
Slices



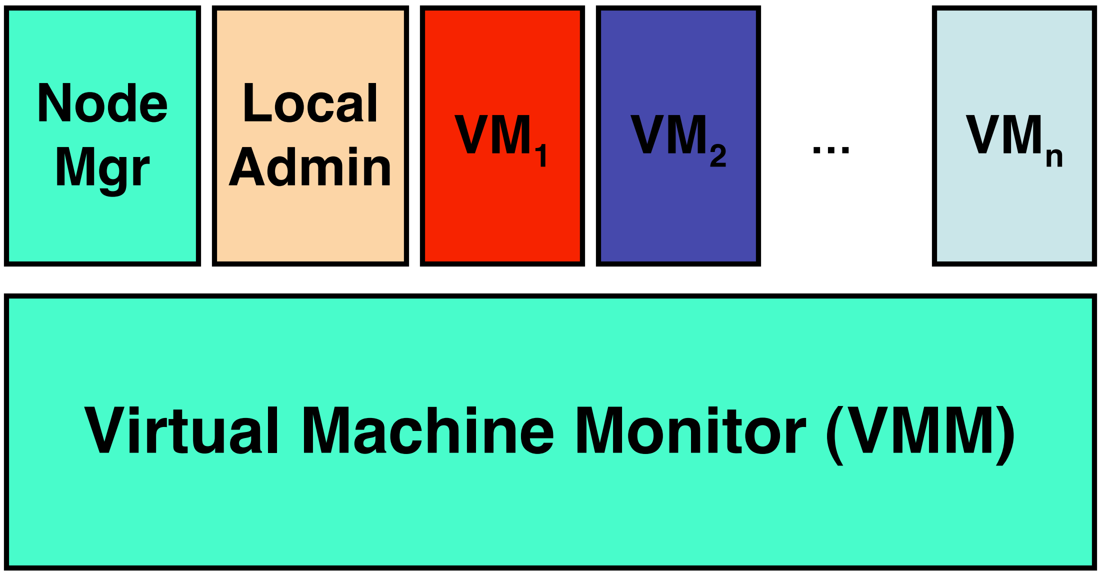
Slices



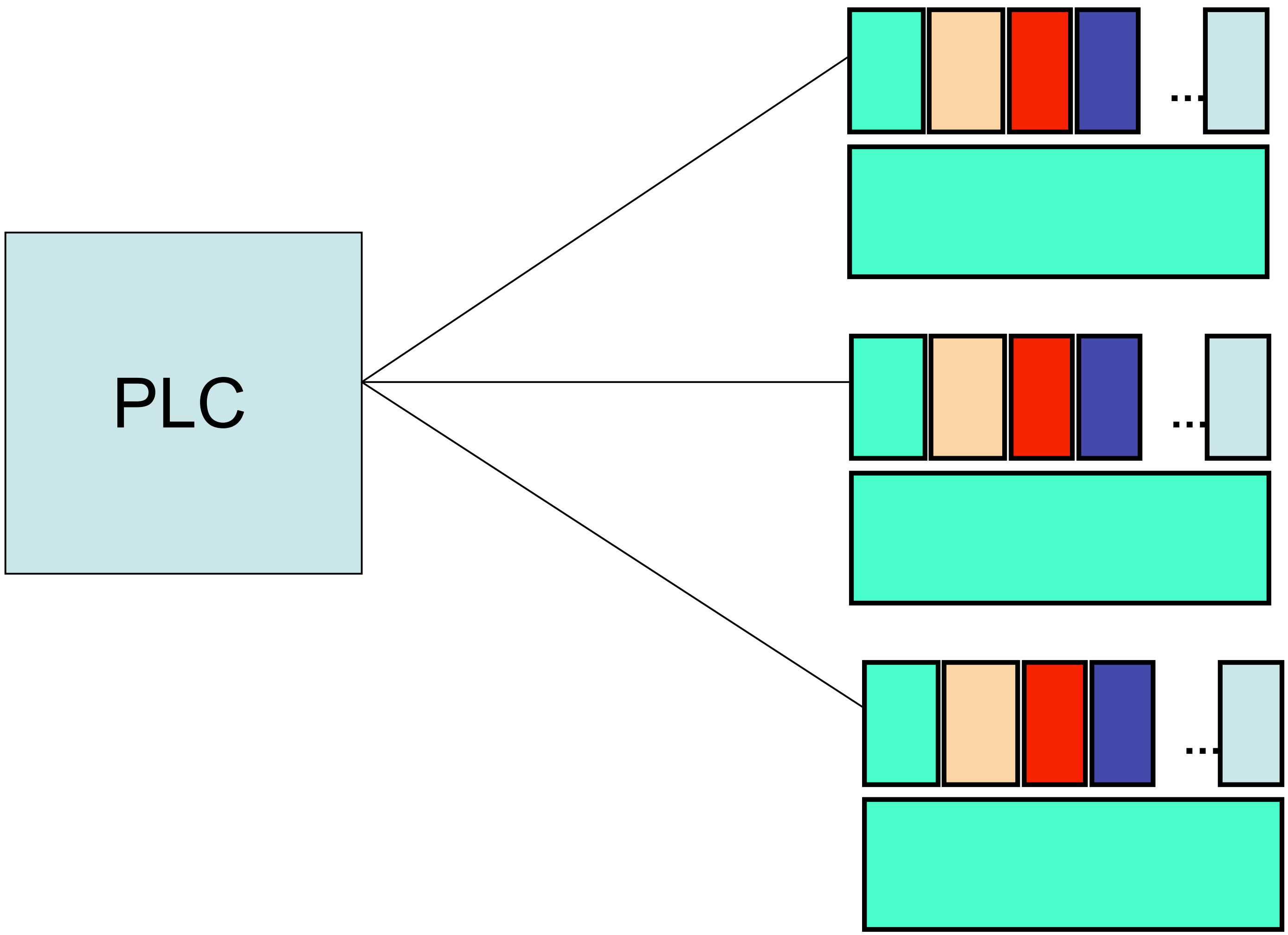
User Opt-in



Per-Node View



Global View



Long-Running Services

- Content Distribution
  - CoDeeN: Princeton
  - Coral: NYU
  - Cobweb: Cornell
- Storage & Large File Transfer
  - LOCI: Tennessee
  - CoBlitz: Princeton
- Anomaly Detection & Fault Diagnosis
  - PIER: Berkeley, Intel
  - PlanetSeer: Princeton
- DHT
  - Bamboo (OpenDHT): Berkeley, Intel
  - Chord (DHash): MIT

Services (cont)

- 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

Services (cont)

- 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

Usage Stats

- Slices: 600+
- Users: 2500+
- Bytes-per-day: 3 - 4 TB
- IP-flows-per-day: 190M
- Unique IP-addrs-per-day: 1M

Two Views of PlanetLab

- Useful research instrument
- Prototype of a new network architecture

- What's interesting about this architecture?
  - more an issue of synthesis than a single clever technique
  - technical decisions that address non-technical requirements

Requirements

1) It must provide a global platform that supports both short-term experiments and long-running services.
   - services must be isolated from each other
   - multiple services must run concurrently
   - must support real client workloads

Requirements

2) It must be available now, even though no one knows for sure what “it” is.
   - deploy what we have today, and evolve over time
   - make the system as familiar as possible (e.g., Linux)
   - accommodate third-party management services

Requirements

3) We must convince sites to host nodes running code written by unknown researchers from other organizations.
   - protect the Internet from PlanetLab traffic
   - must get the trust relationships right

Requirements

4) Sustaining growth depends on support for site autonomy and decentralized control.
   - sites have final say over the nodes they host
   - must minimize (eliminate) centralized control

Requirements

5) It must scale to support many users with minimal resources available.
   - expect under-provisioned state to be the norm
   - shortage of logical resources too (e.g., IP addresses)

Design Challenges

- Develop a management (control) plane that accommodates these often conflicting requirements.
- Balance the need for isolation with the reality of scarce resources.
- Maintain a stable and usable system while continuously evolving it.

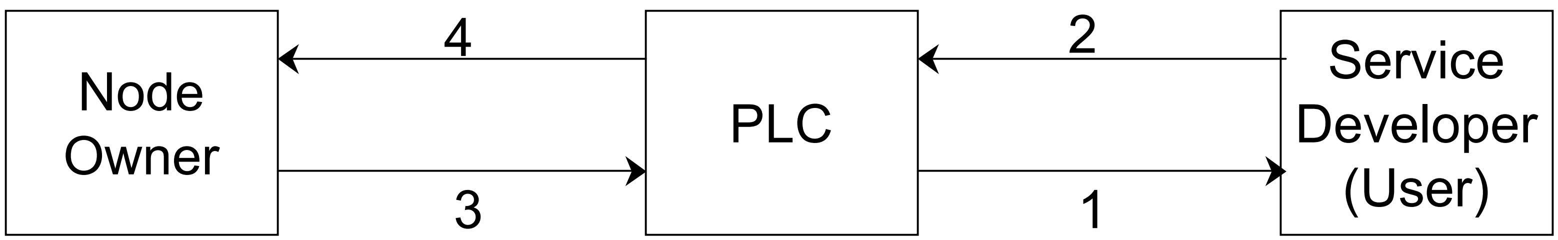
Trust Relationships

Princeton
Berkeley
Washington
MIT
Brown
CMU
NYU
EPFL
Harvard
HP Labs
Intel
NEC Labs
Purdue
UCSD
SICS
Cambridge
Cornell
…

Trusted Intermediary (PLC)

princeton_codeen
nyu_d
cornell_beehive
att_mcash
cmu_esm
harvard_ice
hplabs_donutlab
idsl_psepr
irb_phi
paris6_landmarks
mit_dht
mcgill_card
huji_ender
arizona_stork
ucb_bamboo
ucsd_share
umd_scriptroute
…

Trust Relationships (cont)



1) PLC expresses trust in a user by issuing it credentials to access a slice

2) Users trust PLC to create slices on their behalf and inspect credentials

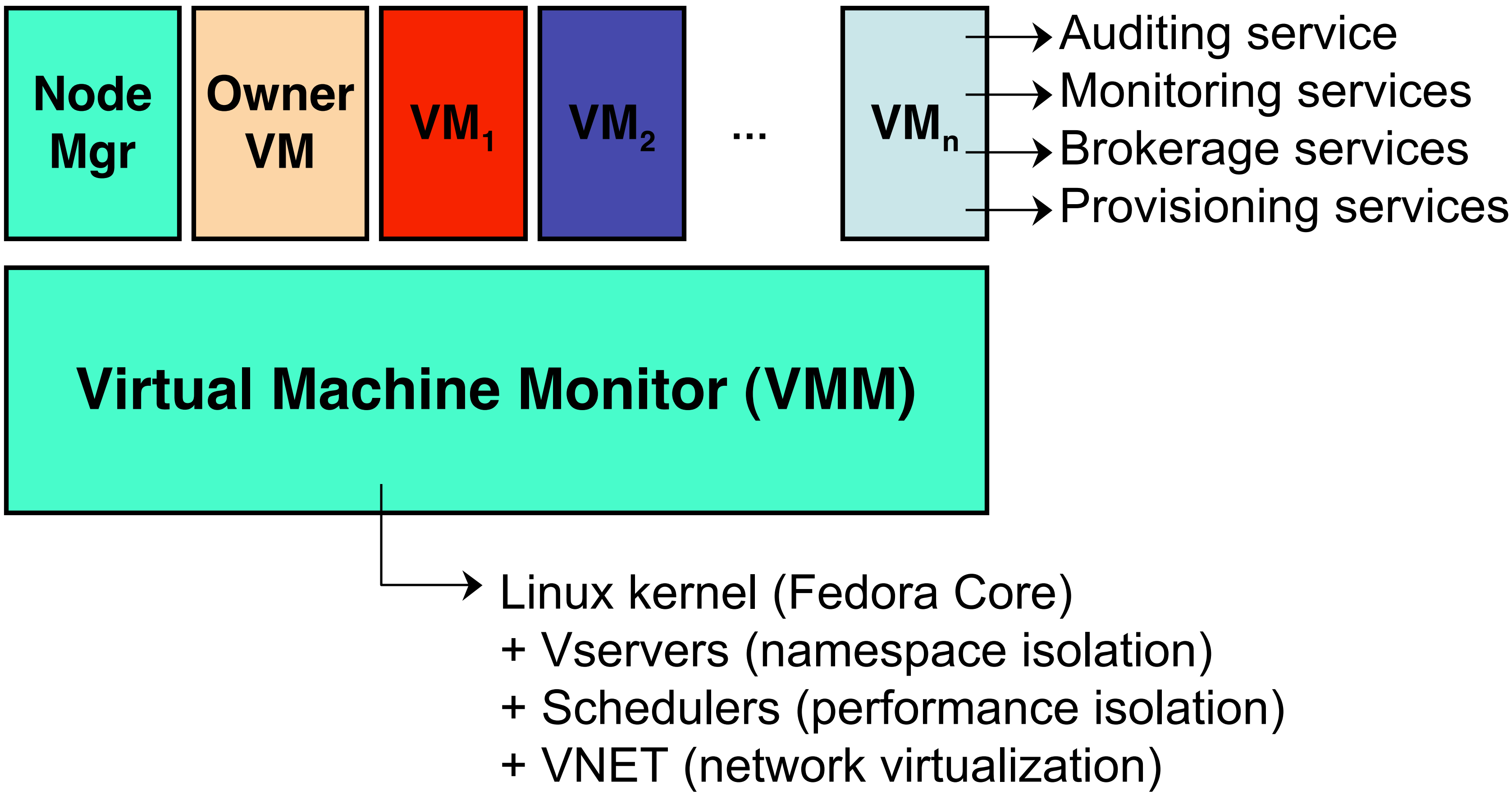
3) Owner trusts PLC to vet users and map network activity to right user

4) PLC trusts owner to keep nodes physically secure

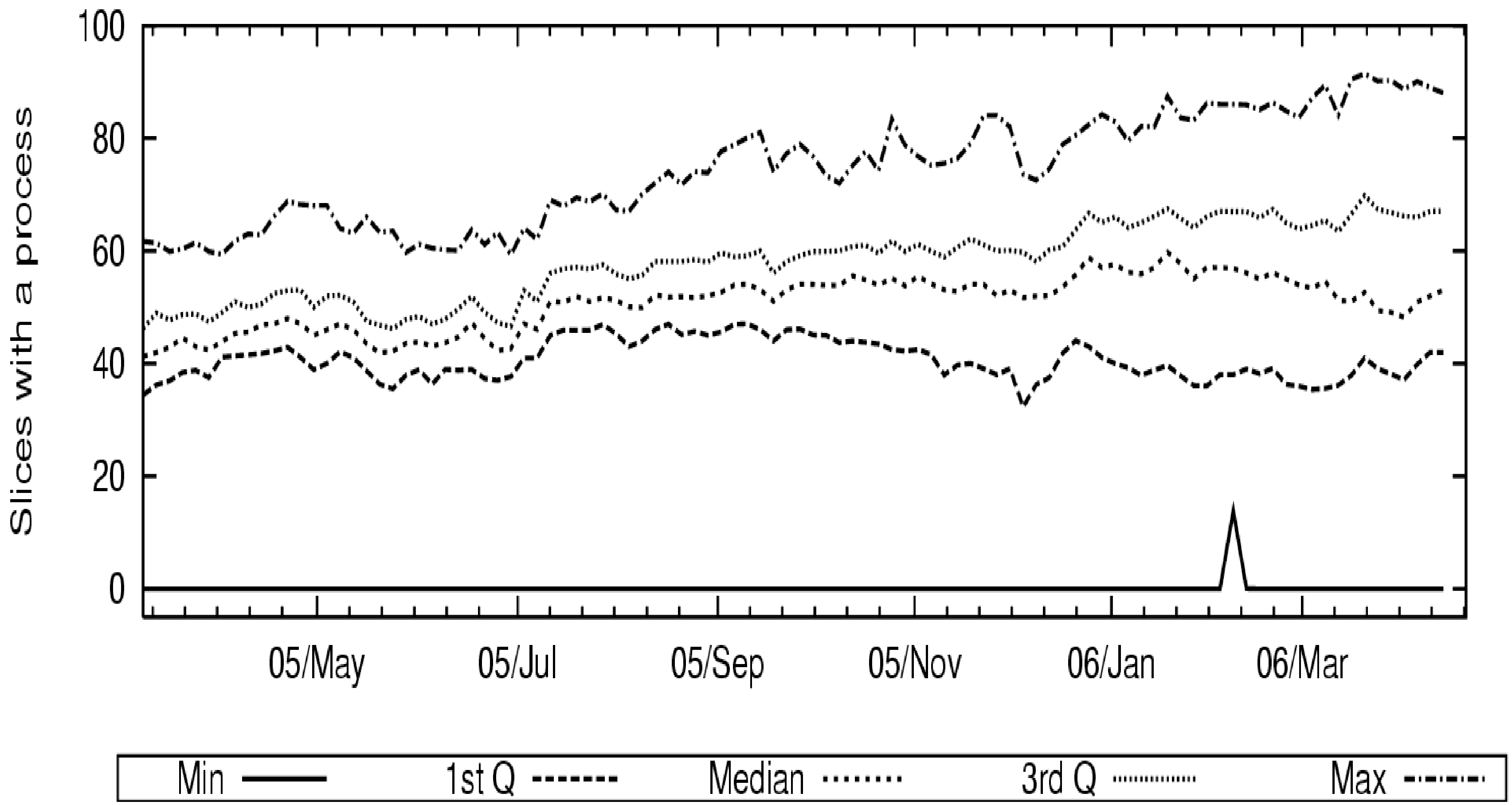
Decentralized Control

- Owner autonomy
  - owners allocate resources to favored slices
  - owners selectively disallow unfavored slices
- Delegation
  - PLC grants tickets that are redeemed at nodes
  - enables third-party management services
- Federation
  - create “private” PlanetLabs using MyPLC
  - establish peering agreements

Virtualization



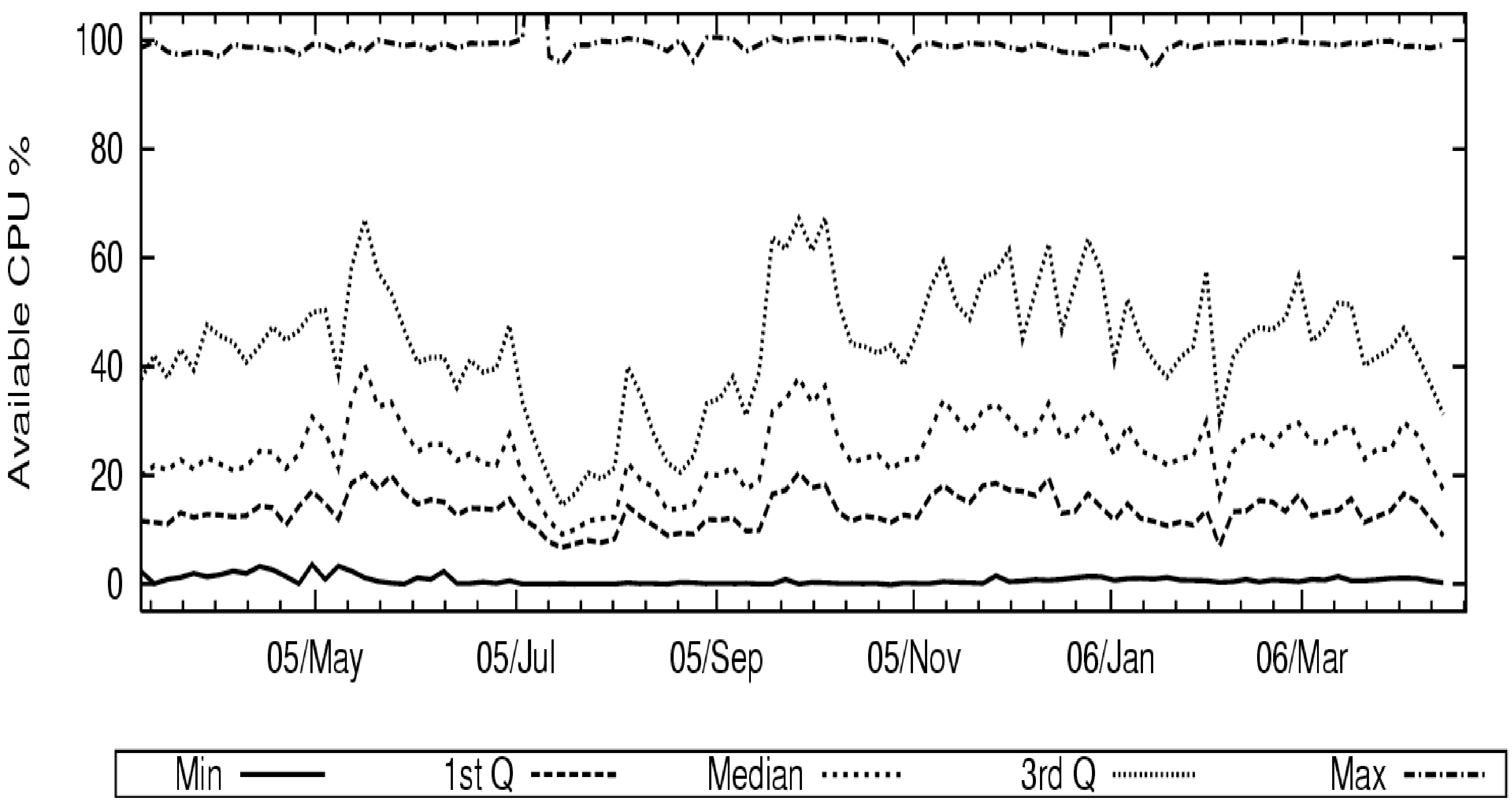
Active Slices



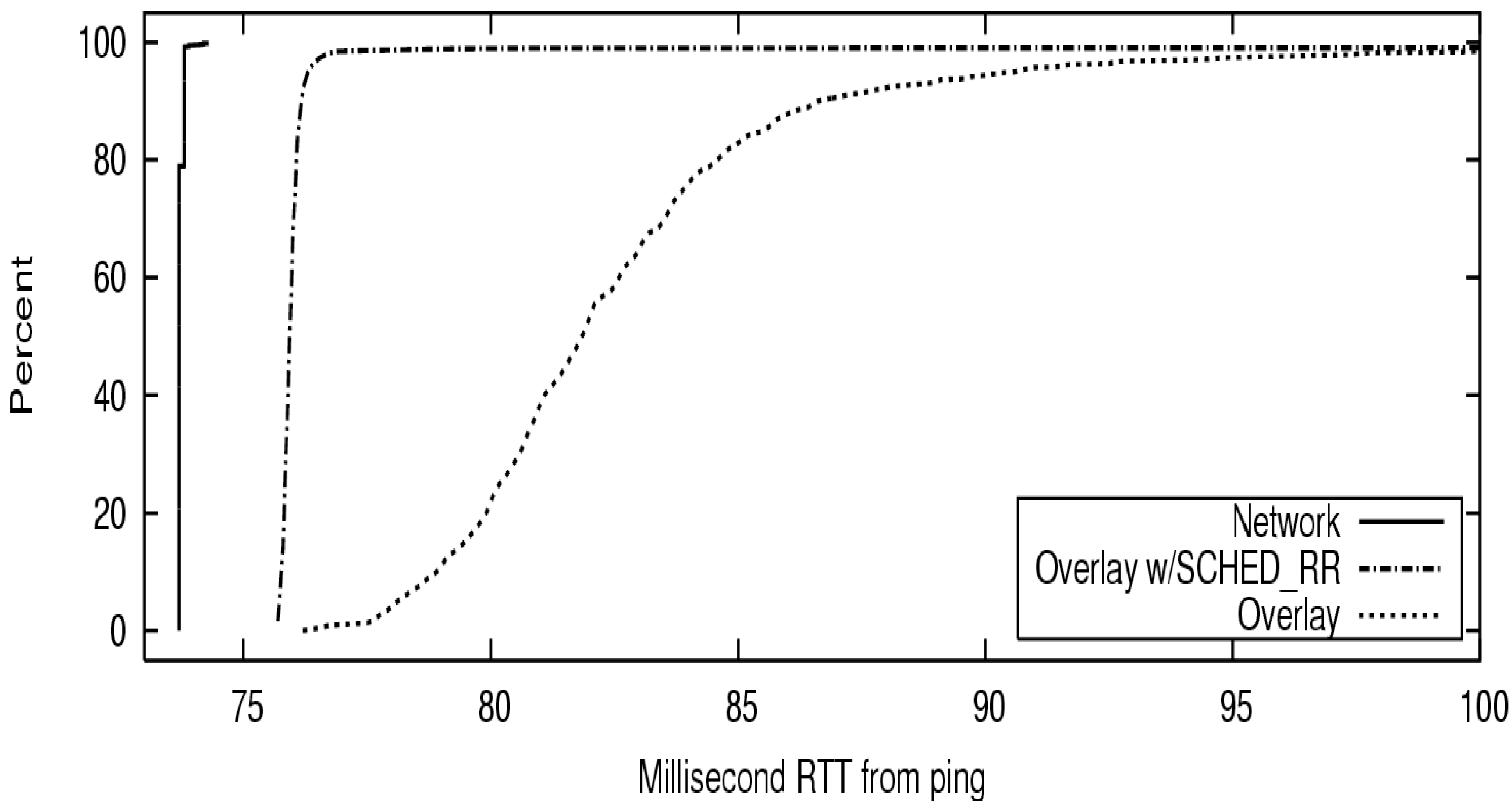
Resource Allocation

- Decouple slice creation and resource allocation
  - given a “fair share” by default when created
  - acquire additional resources, including guarantees
- Fair share with protection against thrashing
  - 1/N*th* of CPU
  - 1/N*th* of link bandwidth
    - owner limits peak rate
    - upper bound on average rate (protect campus bandwidth)
  - disk quota
  - memory limits not practical
    - kill largest user of physical memory when swap at 90%
    - reset node when swap at 95%

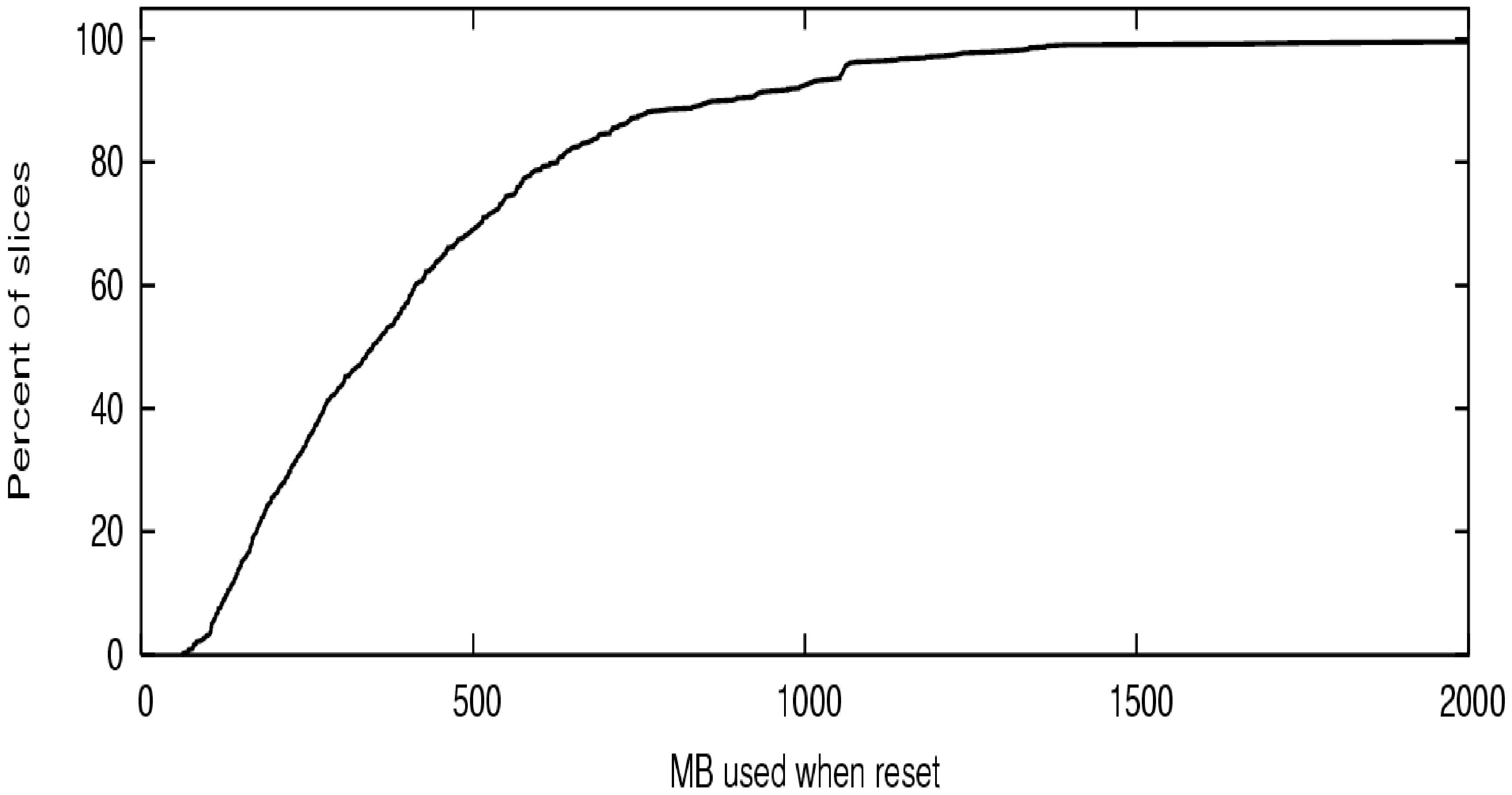
CPU Availability



Scheduling Jitter



Memory Availability



Evolution vs Intelligent Design

- Favor evolution over clean slate
- Favor design principles over a fixed architecture
- Specifically…
  - leverage existing software and interfaces
  - keep VMM and control plane orthogonal
  - exploit virtualization
    - vertical: management services run in slices
    - horizontal: stacks of VMs
  - give no one root (least privilege + level playing field)
  - support federation (divergent code paths going forward)

Other Lessons

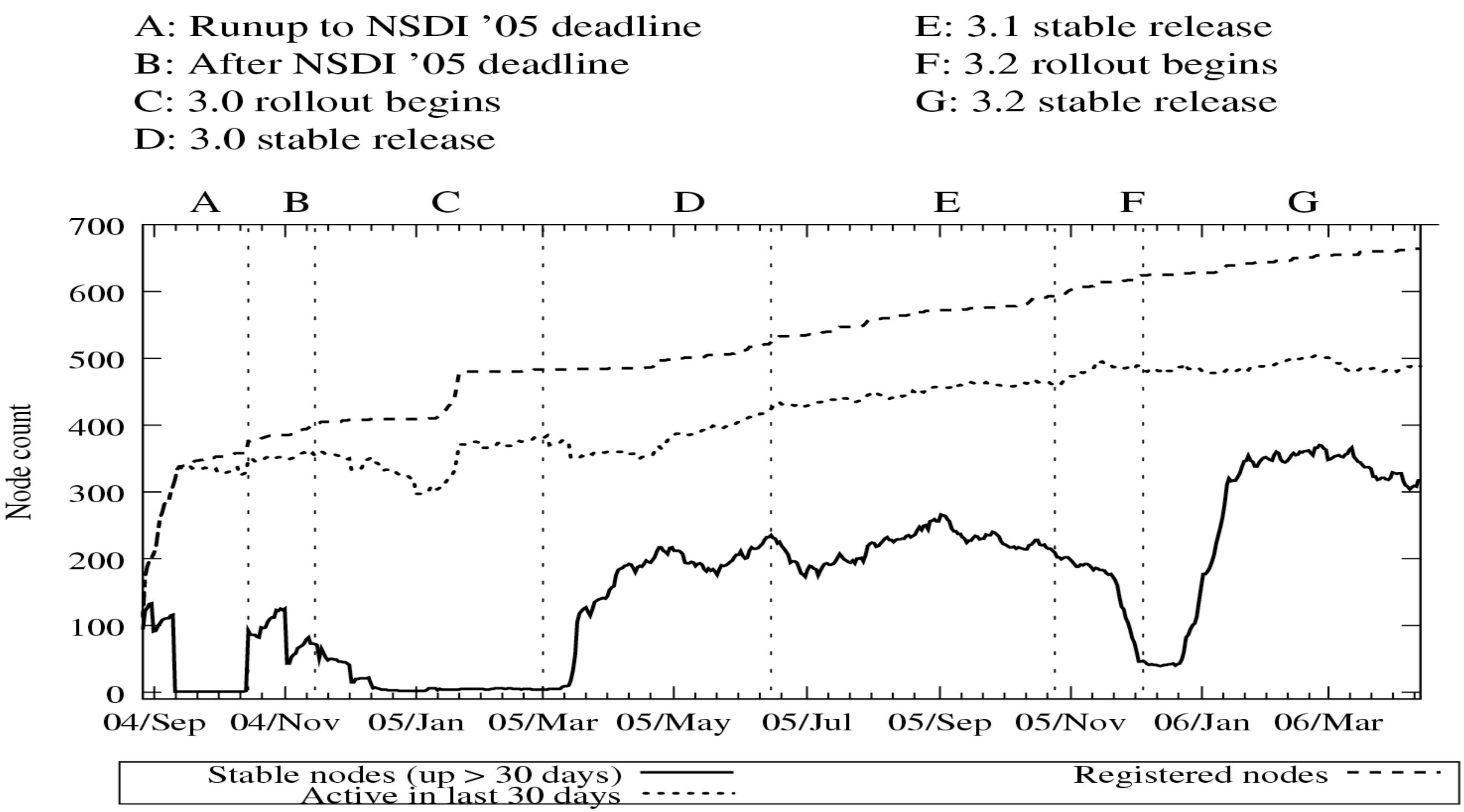
- Inferior tracks lead to superior locomotives
- Empower the user: **`yum`**
- Build it and they (research papers) will come
- Overlays are not networks
- Networks are just overlays
- PlanetLab: We debug your network
- From universal connectivity to gated communities
- If you don't talk to your university's general counsel, you aren't doing network research
- Work fast, before anyone cares

Collaborators

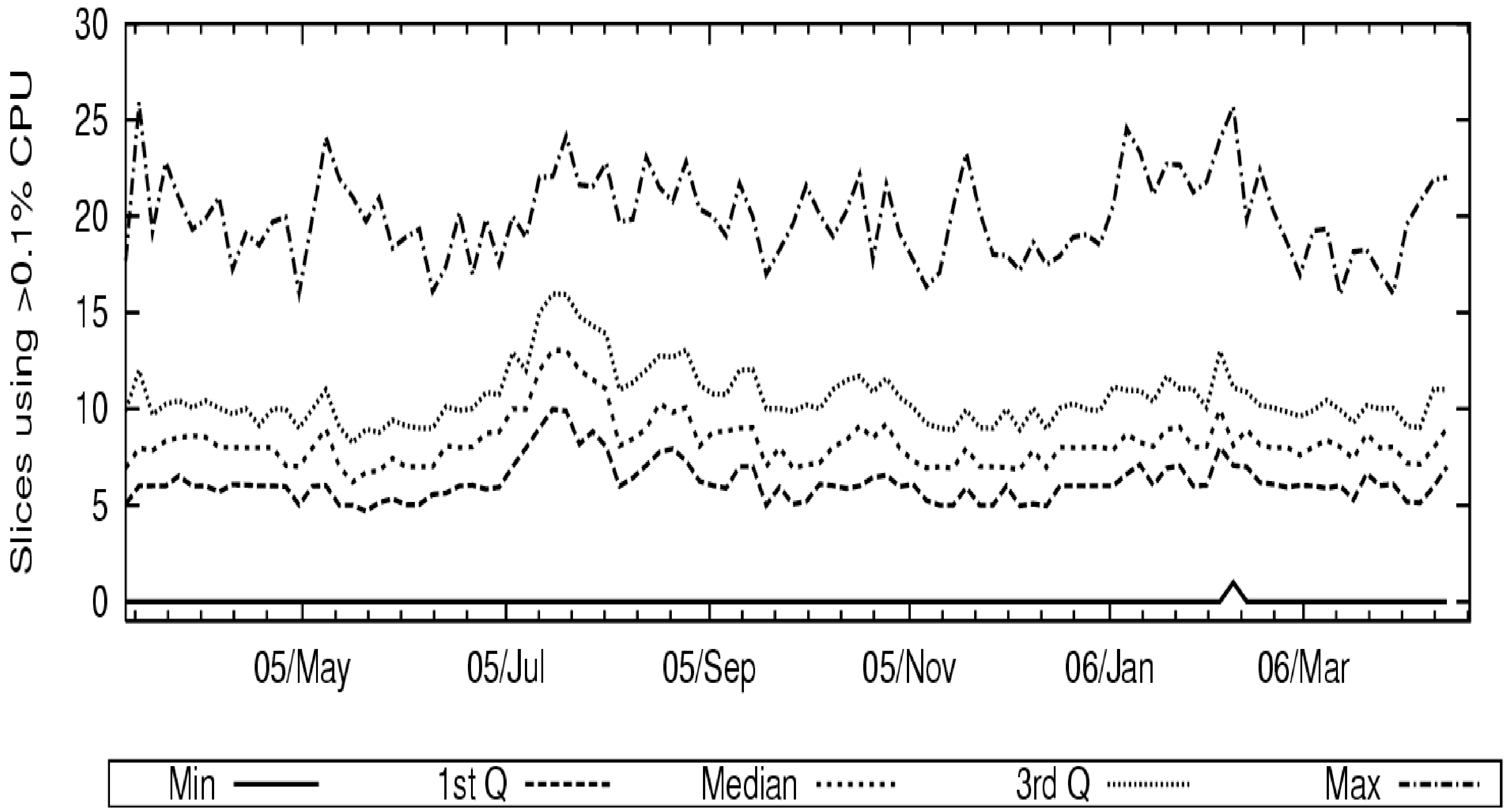
- Andy Bavier
- Marc Fiuczynski
- Mark Huang
- Scott Karlin
- Aaron Klingaman
- Martin Makowiecki
- Reid Moran
- Steve Muir
- Stephen Soltesz
- Mike Wawrzoniak

- David Culler, Berkeley
- Tom Anderson, UW
- Timothy Roscoe, Intel
- Mic Bowman, Intel
- John Hartman, Arizona
- David Lowenthal, UGA
- Vivek Pai, Princeton
- David Parkes, Harvard
- Amin Vahdat, UCSD
- Rick McGeer, HP Labs

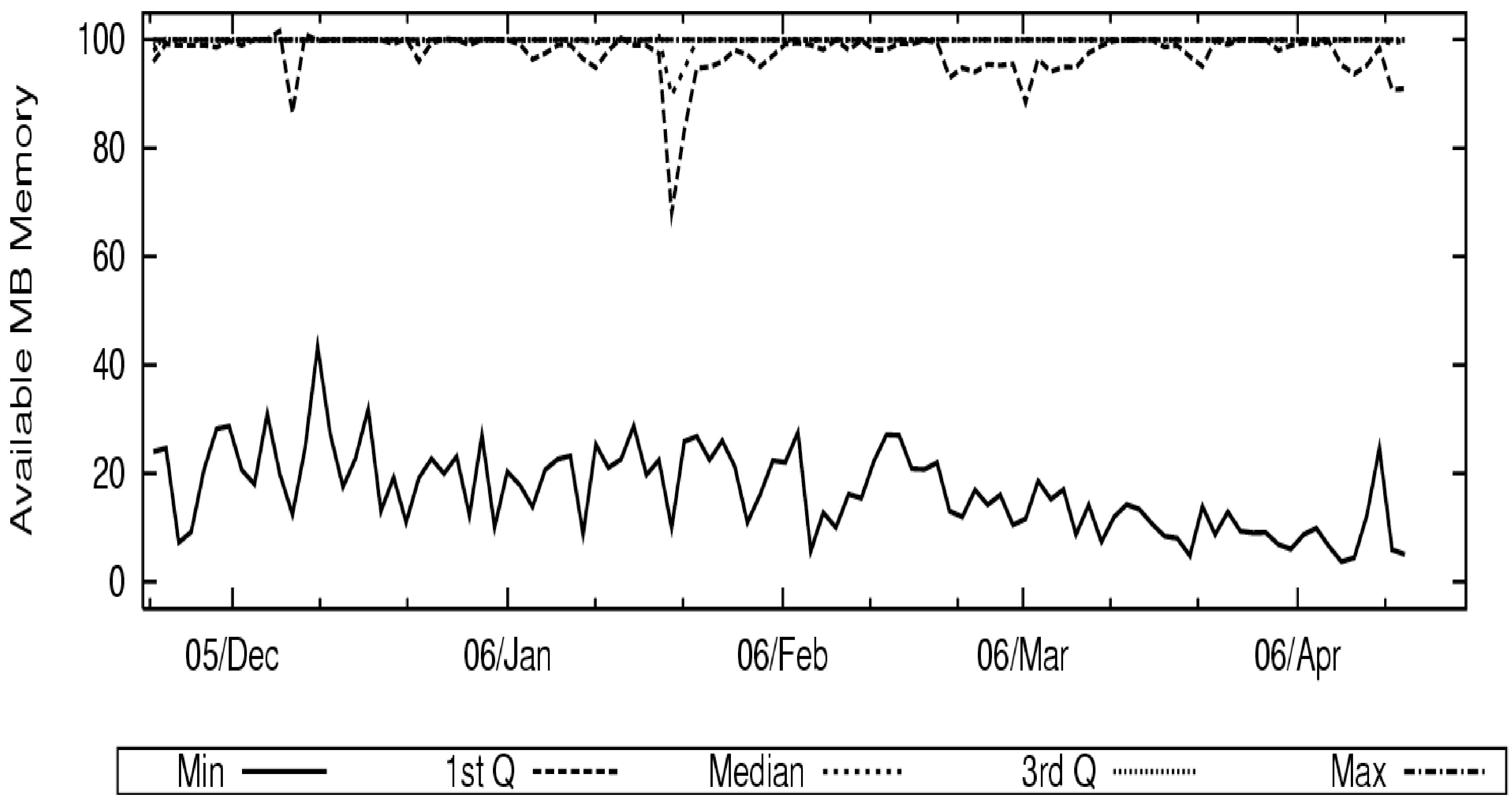
Node Availability



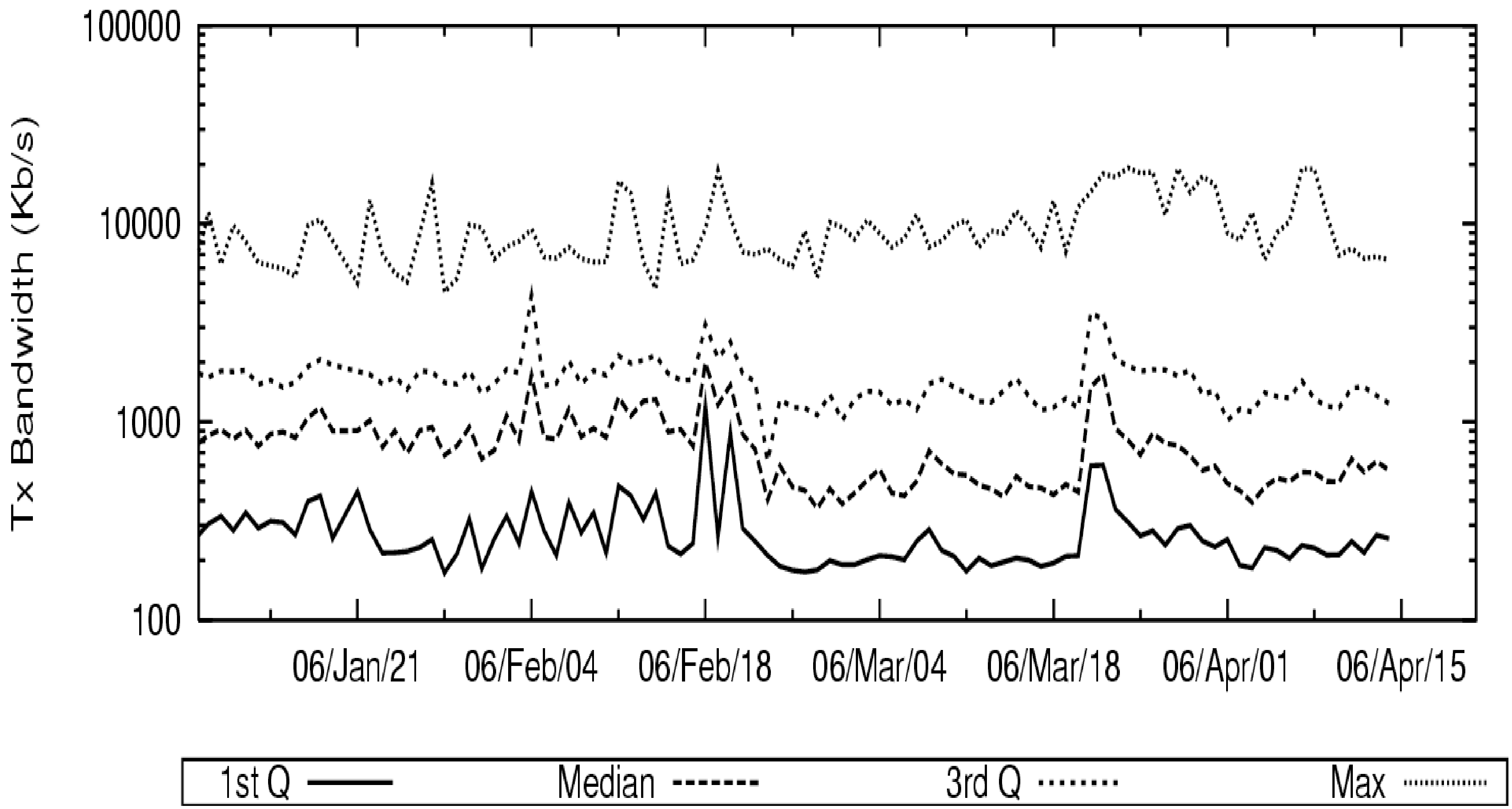
Live Slices


30
25
20
15
10
5
0
Slices using >0.1% CPU
05/May
05/Jul
05/Sep
05/Nov
06/Jan
06/Mar
Min
1st Q
Median
3rd Q
Max

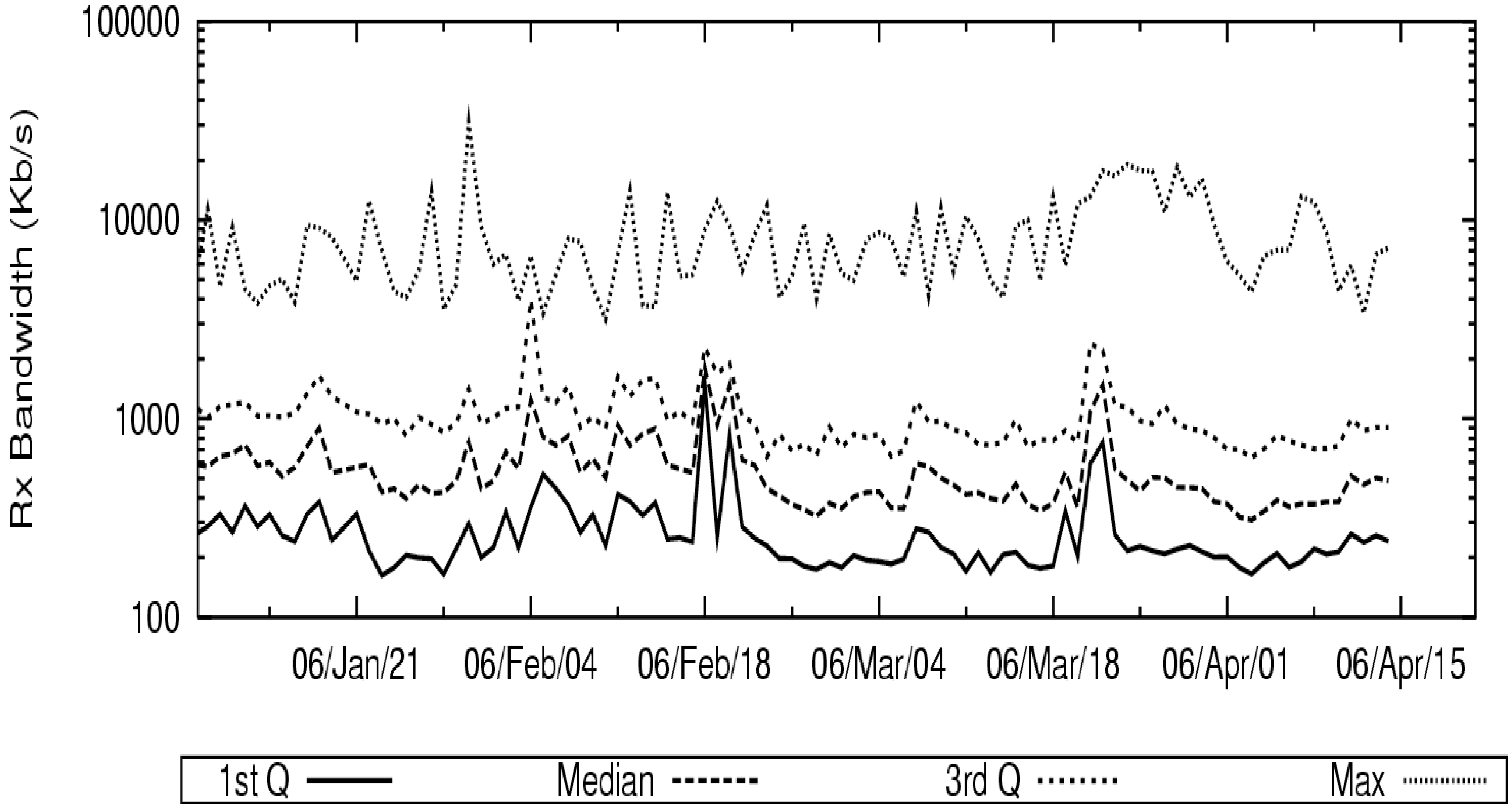
Memory Availability



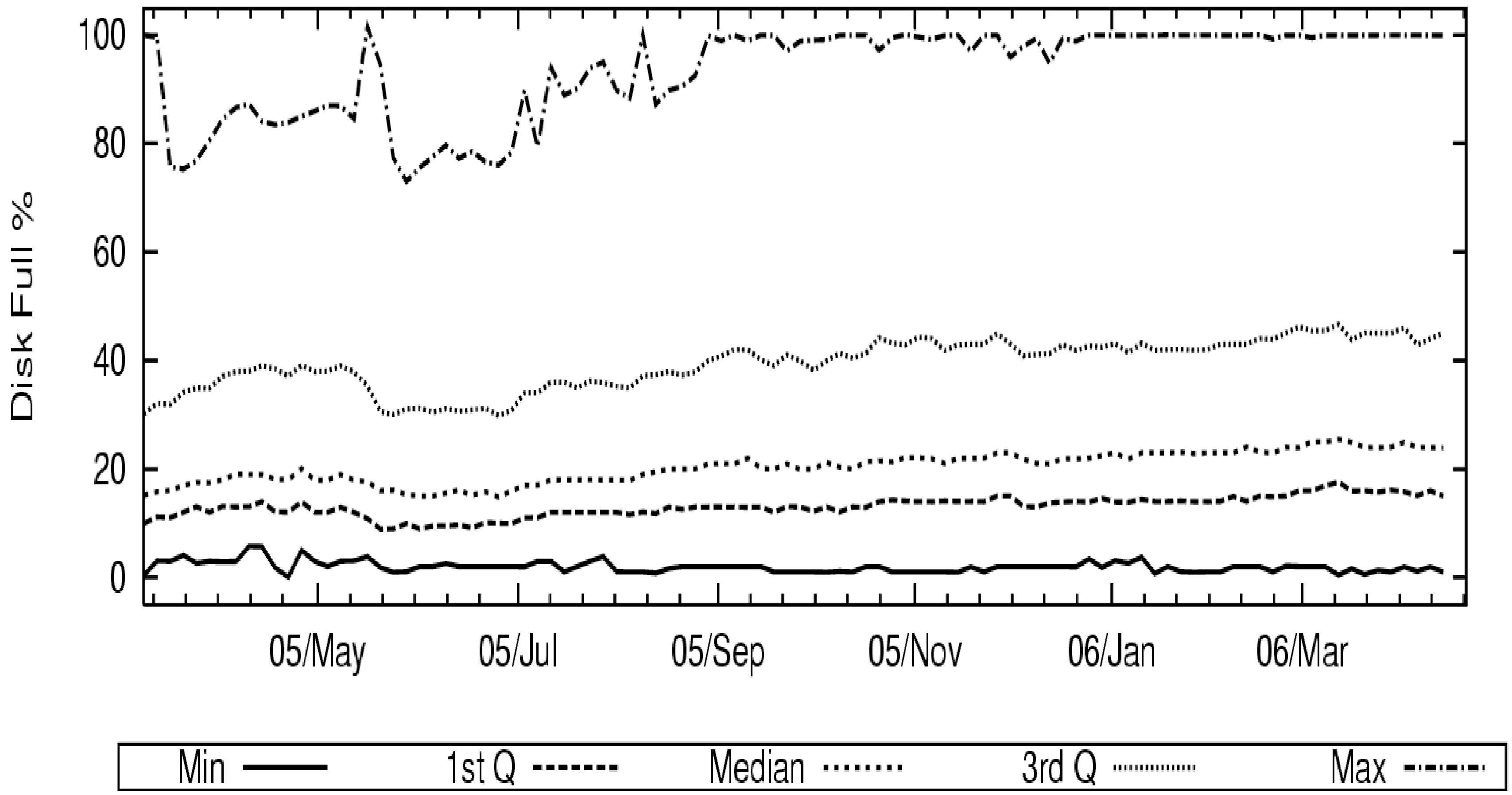
Bandwidth Out



Bandwidth In

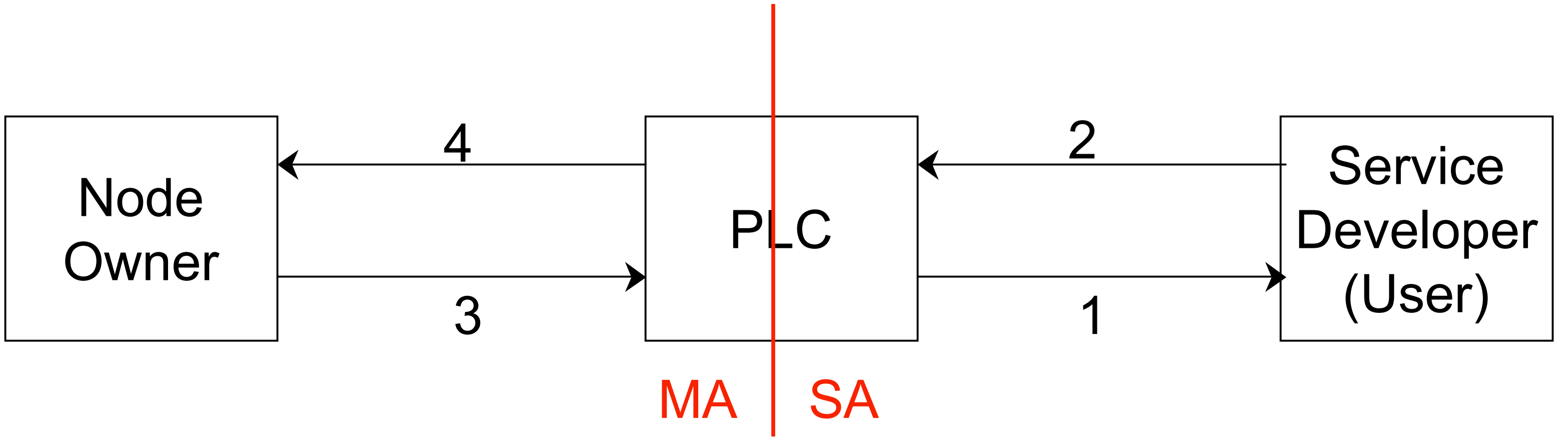


Disk Usage



Disk Full %
100
80
60
40
20
0
05/May
05/Jul
05/Sep
05/Nov
06/Jan
06/Mar
Min
1st Q
Median
3rd Q
Max

Trust Relationships (cont)



1) PLC expresses trust in a user by issuing it credentials to access a slice

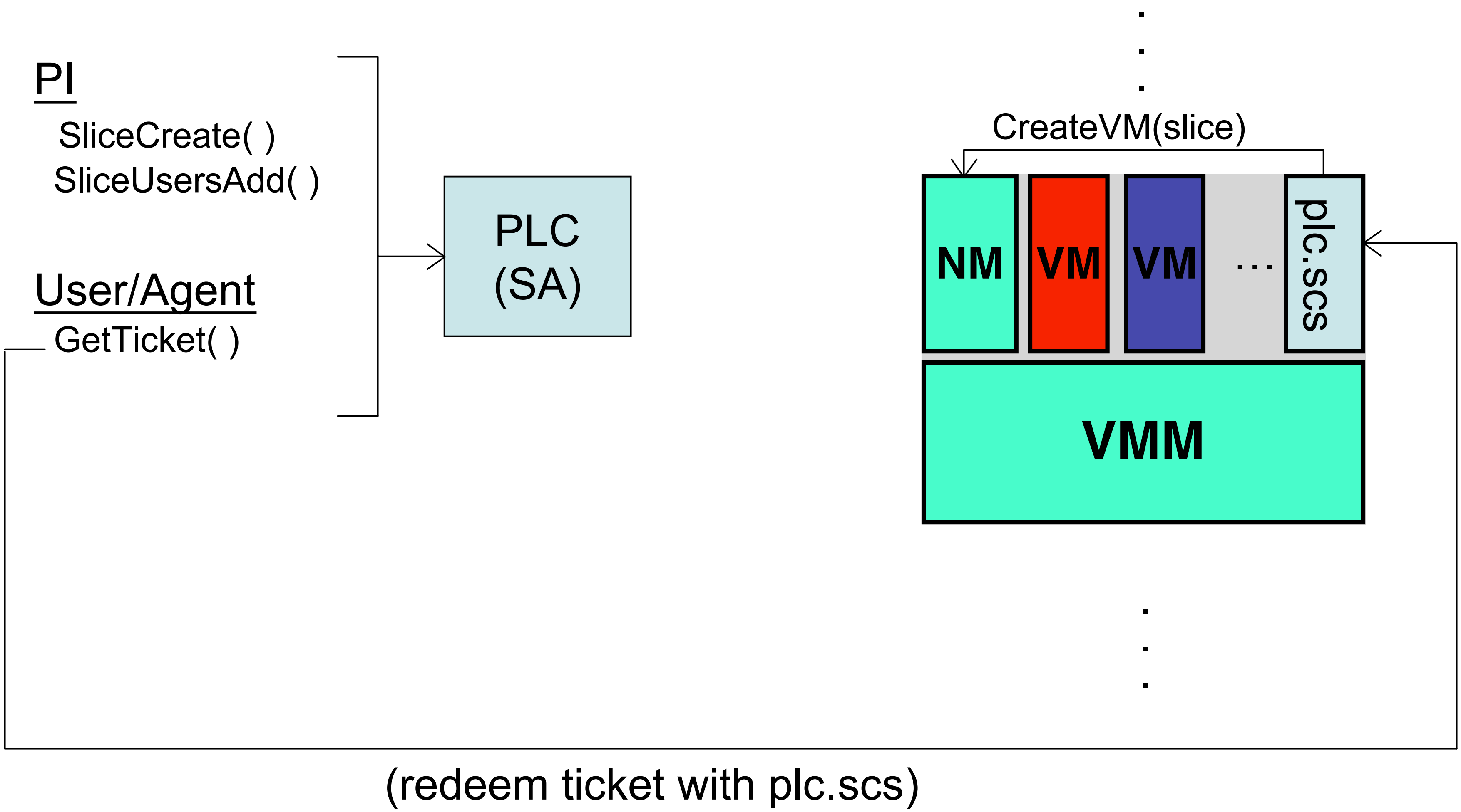
2) Users trust to create slices on their behalf and inspect credentials

3) Owner trusts PLC to vet users and map network activity to right user

4) PLC trusts owner to keep nodes physically secure

MA = Management Authority | SA = Slice Authority

Slice Creation



Brokerage Service