Measurement II: Strategies, Pitfalls, Platforms

Nick Feamster
CS 7260
February 12, 2007

Internet Measurement

- Process of collecting data that measure certain phenomena about the network
  - Should be a science
  - *Today:* closer to an art form
- **Key goal:** Reproducibility
- "Bread and butter" of networking research
  - Deceptively complex
  - Probably one of the most difficult things to do correctly

The Importance of Context: Case Studies with Routing Data

Context Pitfall: AS-Level Topologies

- **Question:** What is the Internet's AS-level topology?
- **Strawman approach**
  - Routeviews routing table dumps
  - Adjacency for each pair of ASes in the AS path

- Problems with the approach?
  - **Completeness:** Many edges could be missing. Why?
    - Single-path routing
    - Policy: ranking and filtering
    - Limited vantage points
  - **Accuracy**
  - **Coarseness**

Context Pitfall: Routing Instability

- **Question:** Does worm propagation cause routing instability?
- **Strawman approach:**
  - Observe routing data collected at RIPE RIRs
  - Correlate routing update traffic in logs with time of worm spread
  - Finding: Lots of routing updates at the time of the worm sprreading!
  - **(Bogus) conclusion:** Worm spreading causes route instability

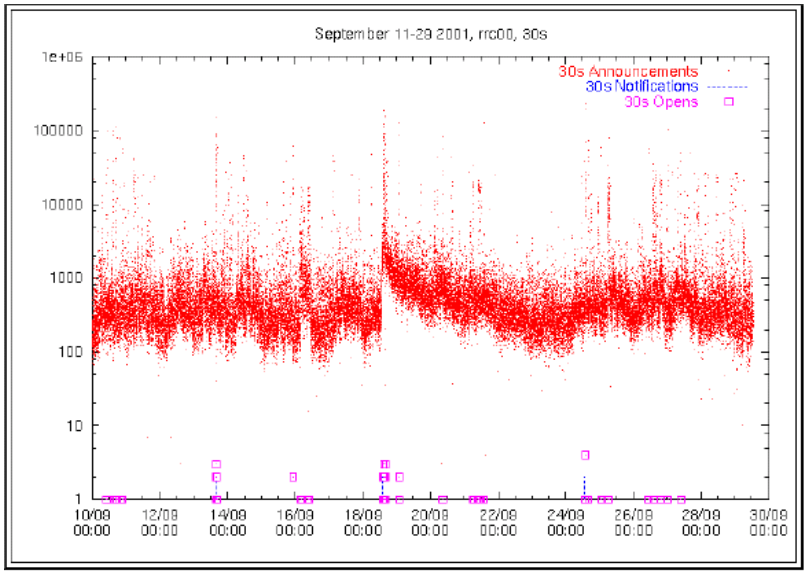


Figure 5: A zoom-in on the BGP message storm of 18–22 September.

Cowie *et al.*, "Global Routing Instabilities Triggered by Code Red II and Nimda Worm Attacks"

Missing/Ignored Context: Instability + eBGP multihop …

Strategy: Examine the Zeroth-Order

- Paxson calls this “looking at spikes and outliers”
- **More general:** Look at the data, not just aggregate statistics
  - Tempting/dangerous to blindly compute aggregates
  - Timeseries plots are telling (gaps, spikes, etc.)
  - Basics
    - Are the raw trace files empty?
      - Need not be 0-byte files (*e.g.,* BGP update logs have state messages but no updates)
    - Metadata/context: Did weird things happen during collection (machine crash, disk full, etc.)

Strategy: Cross-Validation

- Paxson breaks cross validation into two aspects
  - Self-consistency checks (and sanity checks)
  - Independent observations
    - Looking at same phenomenon in multiple ways

- What are some other examples of each of these?

Example Sanity Checks

- Is time moving backwards?
  - PS1's IGP packet trace
  - Paxson's probing example
  - **Typical cause:** Clock synchronization issues
- Has the the speed of light increased?
  - *E.g.*, 10ms cross-country latencies
- Do values make sense?
  - IP addresses that look like 0.0.1.2 indicate bug

BGP Routing Updates: Example

TIME: 07/06/06 19:49:52
TYPE: BGP4MP/STATE_CHANGE
PEER: 18.31.0.51 AS65533
STATE: Active/Connect

TIME: 07/06/06 19:49:52
TYPE: BGP4MP/STATE_CHANGE
PEER: 18.31.0.51 AS65533
STATE: Connect/Opensent

TIME: 07/06/06 19:49:52
TYPE: BGP4MP/STATE_CHANGE
PEER: 18.31.0.51 AS65533
STATE: Opensent/Active

TIME: 07/06/06 19:49:55
TYPE: BGP4MP/MESSAGE/Update
FROM: 18.168.0.27 AS3
TO: 18.7.14.168 AS3
WITHDRAW
12.105.89.0/24
64.17.224.0/21
64.17.232.0/21
66.63.0.0/19
89.224.0.0/14
198.92.192.0/21
204.201.21.0/24

Accuracy issue: Old versions of Zebra would not process updates during a table dump…buggy timestamps.

Cross-Validation Example

- Traceroutes captured in parallel with BGP routing updates
- **Puzzle**
  - Route monitor sees route withdrawal for prefix
  - Routing table has no route to the prefix
  - IP addresses within prefix still reachable from within the IP address space (*i.e.*, traceroute goes through)
- Why?
  - Collection bugs … or
  - Broken mental model of routing setup

Databases: Secret Weapon

- Easy way to get lots of summary statistics
  - Regular first-order stats (*cf.* Paxson's recommendation)
    - Latest timestamp, number of updates, etc.
  - **Cross-validation** becomes easier (quick and dirty SQL)
  - **Joint analysis** of diverse datasets is a common need

- **Caveats!**
  - Insertion must be done properly
    - Always, always save raw data
  - Beware the table join

Horror Story #1: Buggy Postprocessing

- Logs maintained at each host
- Files collected and merged to compute one-way delays

Example RON Monitoring Logs

```
1103659228.224614 S 14b13270 0 8 18.7.14.168 66.26.83.103
1103659228.252509 R 14b13270 1 8 18.7.14.168 66.26.83.103
1103659229.388441 S 55a4b9a1 0 8 18.7.14.168 192.249.24.10
1103659229.611096 R 55a4b9a1 1 8 18.7.14.168 192.249.24.10
1103659231.200177 S bf1207a0 0 8 18.7.14.168 12.46.129.20
1103659231.270053 R bf1207a0 1 8 18.7.14.168 12.46.129.20
1103659233.109900 S 55e244c0 0 8 18.7.14.168 112.12.8.0
1103659234.308722 S 8ba24c76 0 8 18.7.14.168 18.97.168.219
```

- If corresponding ends of logfile missing: set receive time to zero.

"Does the extra effort matter?" (Paxson)

- What if the log files don't match up in time properly?
- What about missing log files?

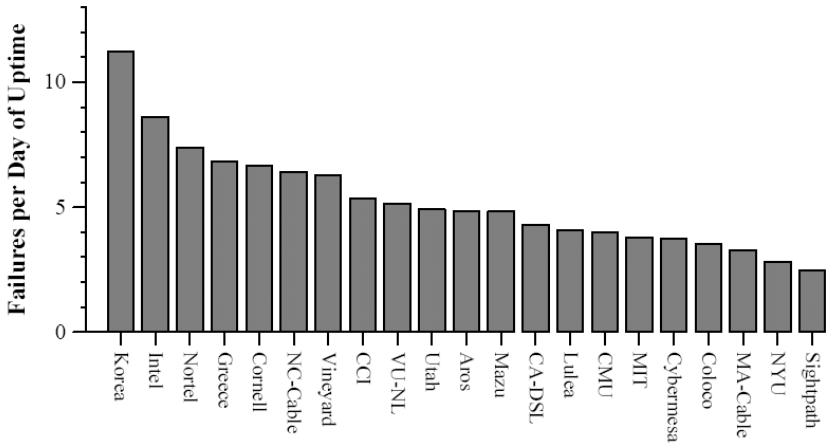
Horror Story #2: Buggy Insertion

- Raw files pulled to archive
  - Archive stores directories month-by-month
  - Files named by unixtime (start,end)
  - Files pulled into directory by month
- Insertion into DB: one archive directory at a time

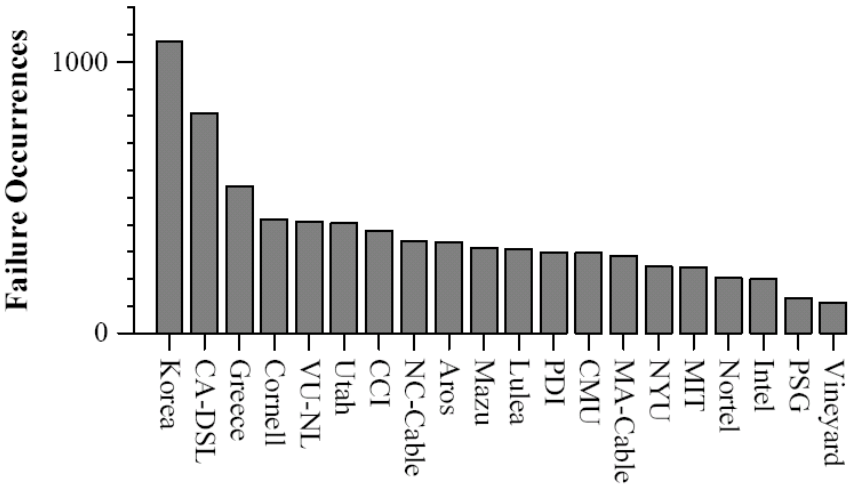
- **Question:** What about files that traverse multiple months?

Horror Story #3: Join Queries

Final Version



Submitted Version



select srchost.name, dsthost.name, count(*) from hosts as srchost, hosts as dsthost, outages where srchost.ip=outages.src **and dsthost.ip=outages.dst …**

Anonymization

- Similar questions arise here as with accuracy
- Researchers always want full packet captures with payloads
  - …but many questions can be answered *without* complete information

- Privacy / de-anonymization issues

Longitudinal Studies

- Extremely valuable
- Extremely hard to maintain
  - Requires constant babysitting (disks fill up, programs/OSes upgraded, IP addresses change, etc.)
- A few pointers
  - Store all mappings that are not invariant
  - Regular regression, backup, first-order stats
  - Paxson's "master script" idea can help with regression

Passive Measurement

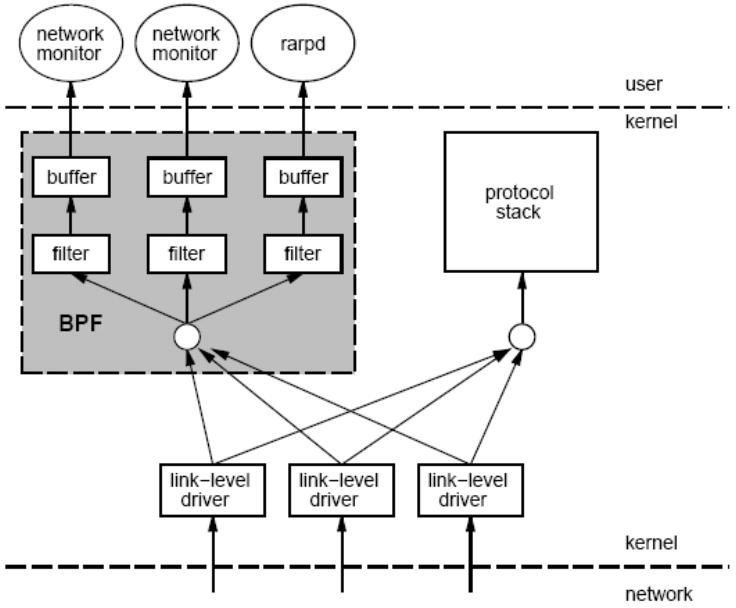
Two Main Approaches

- Packet-level Monitoring
  - Keep packet-level statistics
  - Examine (and potentially, log) variety of packet-level statistics. Essentially, anything in the packet.
  - **Timing**

- Flow-level Monitoring
  - Monitor packet-by-packet (though sometimes sampled)
  - Keep aggregate statistics on a flow

Packet Capture: tcpdump/bpf

- Put interface in promiscuous mode
- Use bpf to extract packets of interest



Accuracy Issues

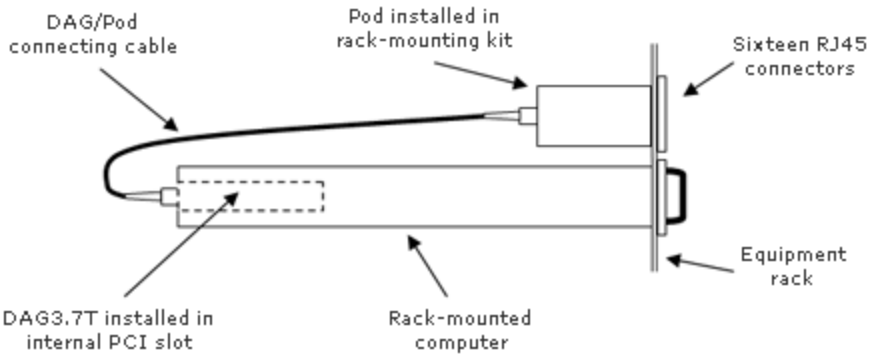
- Packets may be dropped by filter
  - Failure of tcpdump to keep up with filter
  - Failure of filter to keep up with dump speeds

Question: How to recover lost information from packet drops?

Packet Capture on High-Speed Links

Example: Georgia Tech OC3Mon

- Rack-mounted PC
- Optical splitter
- Data Acquisition and Generation (DAG) card



Source: endace.com

Traffic Flow Statistics

- *Flow monitoring* (*e.g.*, Cisco Netflow)
  - Statistics about groups of related packets (*e.g.,* same IP/TCP headers and close in time)
  - Recording header information, counts, and time

- More detail than SNMP, less overhead than packet capture
  - Typically implemented directly on line card

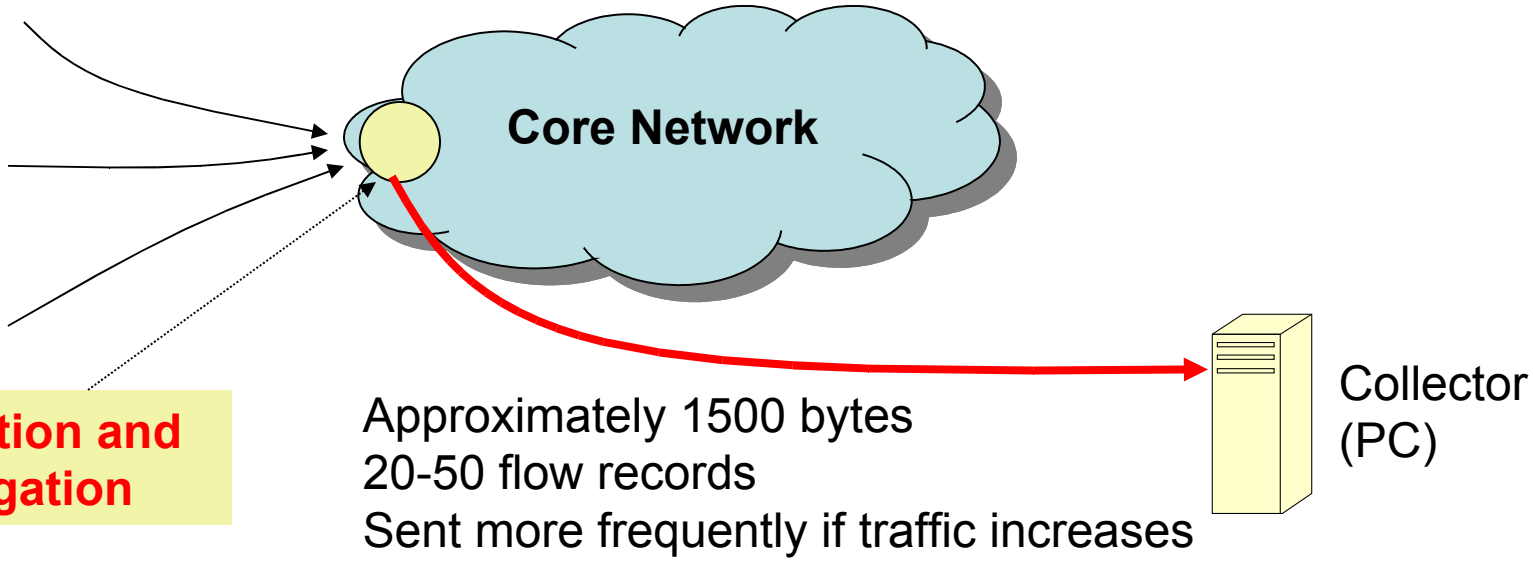
What is a flow?

- **Source IP address**
- **Destination IP address**
- **Source port**
- **Destination port**
- **Layer 3 protocol type**
- TOS byte (DSCP)
- Input logical interface (ifIndex)

Cisco Netflow

- Basic output: “Flow record”
  - Most common version is v5
  - Latest version is v10 (RFC 3917)

- Current version (10) is being standardized in the IETF (*template-based*)
  - More flexible record format
  - Much easier to add new flow record types



Flow Record Contents

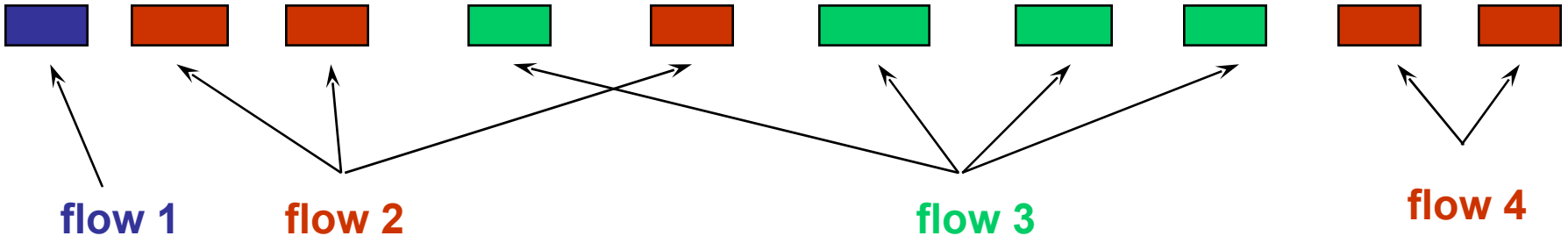
Basic information about the flow…

- Source and Destination, IP address and port
- Packet and byte counts
- Start and end times
- ToS, TCP flags

…plus, information related to routing

- Next-hop IP address
- Source and destination AS
- Source and destination prefix

Aggregating Packets into Flows



- **Criteria 1:** Set of packets that "belong together"
  - Source/destination IP addresses and port numbers
  - Same protocol, ToS bits, …
  - Same input/output interfaces at a router (if known)

- **Criteria 2:** Packets that are "close" together in time
  - Maximum inter-packet spacing (e.g., 15 sec, 30 sec)
  - **Example:** flows 2 and 4 are different flows due to time

Netflow Processing

1. Create and update flows in NetFlow Cache

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	SrcPort	SrcMsk	SrcAS	DstPort	DstMsk	DstAS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1745	4
Fa1/0	173.100.3.2	Fa0/0	10.0.227.12	6	40	0	2491	15	/26	196	15	/24	15	10.0.23.2	740	41.5	1
Fa1/0	173.100.20.2	Fa0/0	10.0.227.12	11	80	10	10000	00A1	/24	180	00A1	/24	15	10.0.23.2	1428	1145.5	3
Fa1/0	173.100.6.2	Fa0/0	10.0.227.12	6	40	0	2210	19	/30	180	19	/24	15	10.0.23.2	1040	24.5	14

1. Expiration

- Inactive timer expired (15 sec is default)
- Active timer expired (30 min (1800 sec) is default)
- NetFlow cache is full (oldest flows are expired)
- RST or FIN TCP Flag

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	SrcPort	SrcMsk	SrcAS	DstPort	DstMsk	DstAS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1800	4

1. Aggregation?

No

Yes

e.g. Protocol-Port Aggregation Scheme becomes

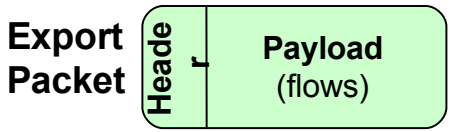
Protocol	Pkts	SrcPort	DstPort	Bytes/Pkt
11	11000	00A2	00A2	1528

1. Export Version

Non-Aggregated Flows – export **Version 5 or 9**

Aggregated Flows – export **Version 8 or 9**

1. Transport Protocol

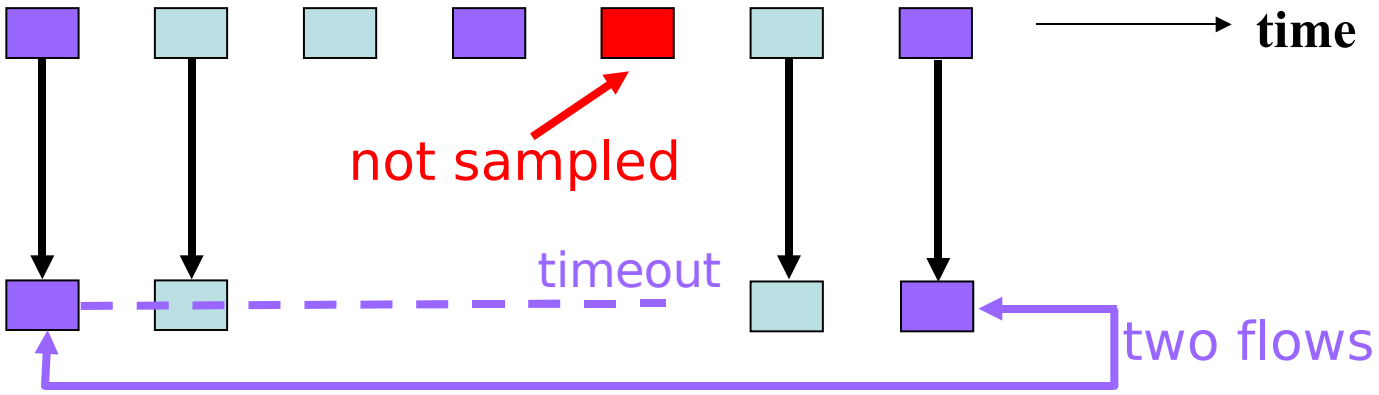


Reducing Measurement Overhead

- **Filtering:** on interface
  - destination prefix for a customer
  - port number for an application (e.g., 80 for Web)
- **Sampling:** before insertion into flow cache
  - Random, deterministic, or hash-based sampling
  - 1-out-of-n or stratified based on packet/flow size
  - *Two types:* packet-level and flow-level
- **Aggregation:** after cache eviction
  - packets/flows with same next-hop AS
  - packets/flows destined to a particular service

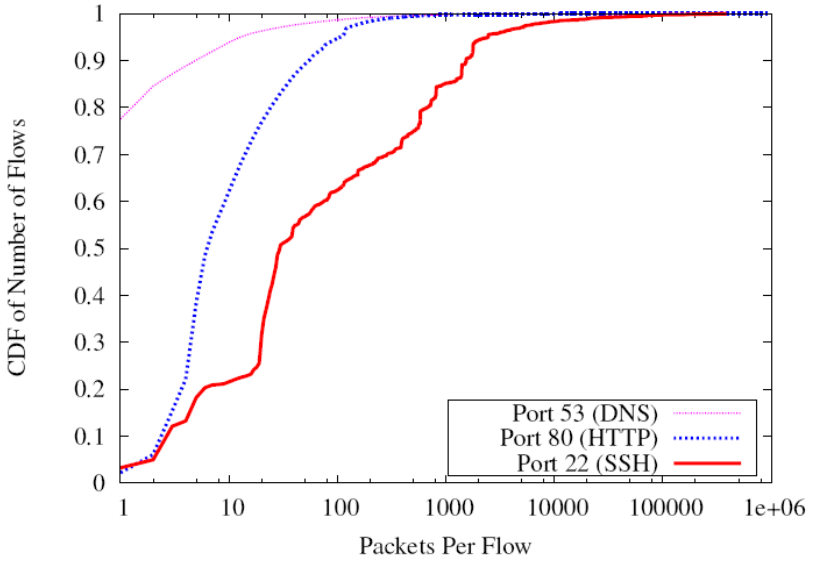
Packet Sampling

- Packet sampling before flow creation (Sampled Netflow)
  - 1-out-of-m sampling of individual packets (*e.g.*, m=100)
  - Create of flow records over the sampled packets
- Reducing overhead
  - Avoid per-packet overhead on (m-1)/m packets
  - Avoid creating records for a large number of small flows
- Increasing overhead (in some cases)
  - May split some long transfers into multiple flow records
  - … due to larger time gaps between successive packets



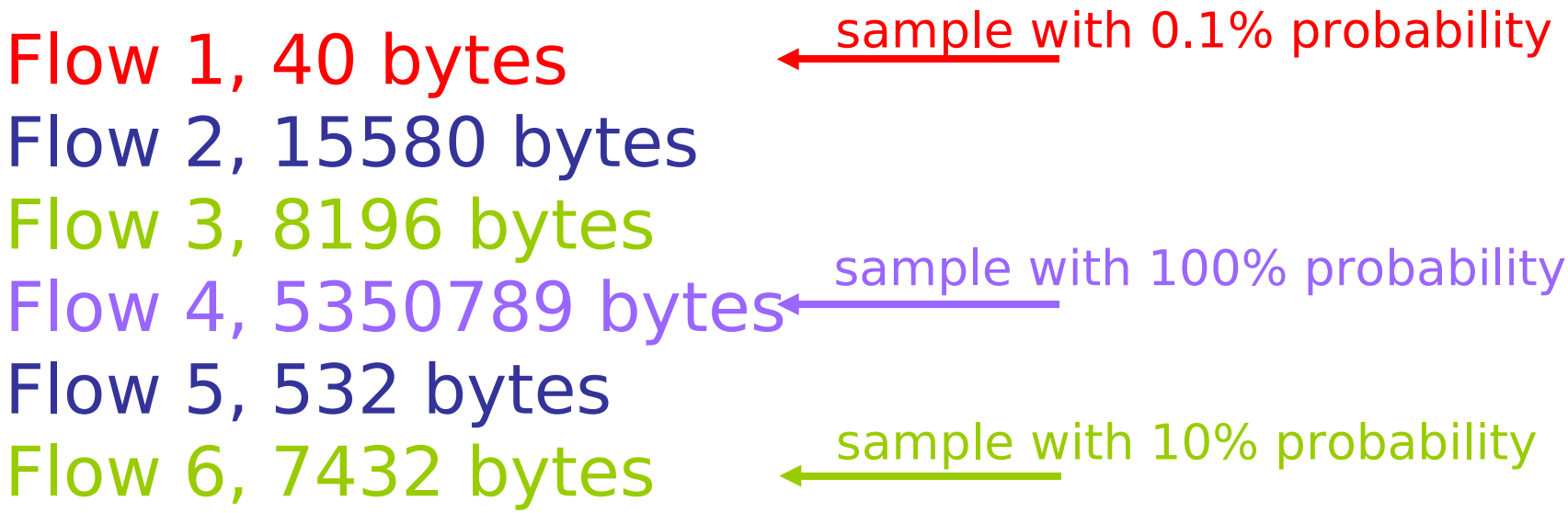
Problems with Packet Sampling

- Determining size of original flows is tricky
  - For a flow originally of size *n*, the size of the *sampled* flow follows a binomial distribution
  - Extrapoliation can result in big errors
  - Much research in reducing such errors (upcoming lectures)
- Flow records can be lost
- Small flows may be eradicated entirely



Sampling: Flow-Level Sampling

- Sampling of flow records evicted from flow cache
  - When evicting flows from table or when analyzing flows
- Stratified sampling to put weight on “heavy” flows
  - Select all long flows and sample the short flows
- Reduces the number of flow records
  - Still measures the vast majority of the traffic

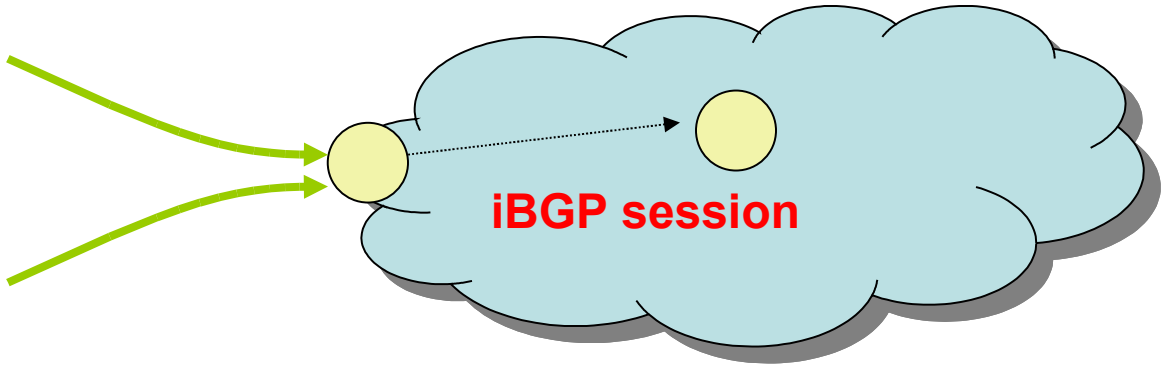


Accuracy Depends on Phenomenon

- Even naïve random sampling probably decent for capturing the *existence* of large flows

- Accurately measuring other features may require different approaches
  - Sizes of large flows
  - Distribution of flow sizes
  - Existence of small flows (coupon collection)
  - Size of small flows
  - Traffic “matrix”

Routing Data



- IGP
- BGP
  - Collection methods
    - eBGP (typically “multihop”)
    - iBGP
  - Table dumps: Periodic, complete routing table state (direct dump from router)
  - Routing updates: Continuous, incremental, best route only

Evaluation Strategies and Platforms

Other Measurement Tools

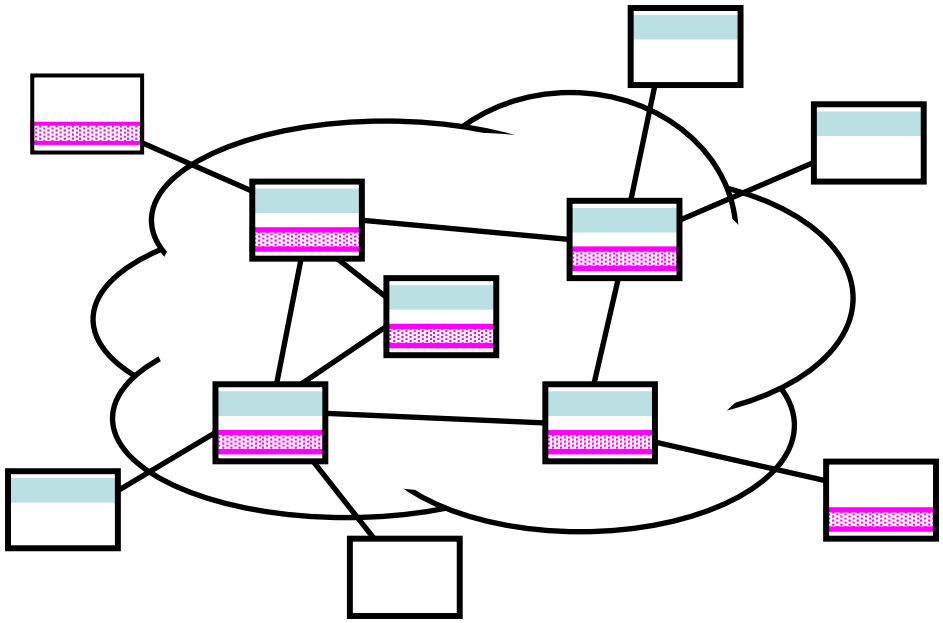
- Scriptroute (http://www.scriptroute.org/)
  - Write new probing tools/techniques, etc.
  - More on PS 2

Evaluation Strategies

- **Simulation**
  - Ns2, SSFNet
  - Advantages: Control
- **Emulation**
  - Emulab
  - Advantages: Real software, more realistic conditions
- **Wide-area Deployment**
  - VINI
  - Simultaneous operation, sharing
  - Advantages: Ability to carry real traffic

Next Lecture: **Comparisons of these different evaluation strategies**

PlanetLab: Distributed Services



Key challenge: Isolation

- **Slice:** Set of VMs are treated as a single entity (*distributed virtualization*)
- Isolation at system call level (vservers)
  - Shared filesystem, memory, etc.
- Network virtualization: safe raw sockets
  - Must be bound to a specific port

Virtualization

- Advantages
  - Simultaneous access to shared physical resources
- Disadvantages
  - Requires scheduling
  - Not running on “raw” hardware. May not see similar performance as the “real” network/system

PlanetLab for Network Measurement

- Nodes are largely at academic sites
  - Other alternatives: RON testbed (*disadvantage:* difficult to run long running measurements)

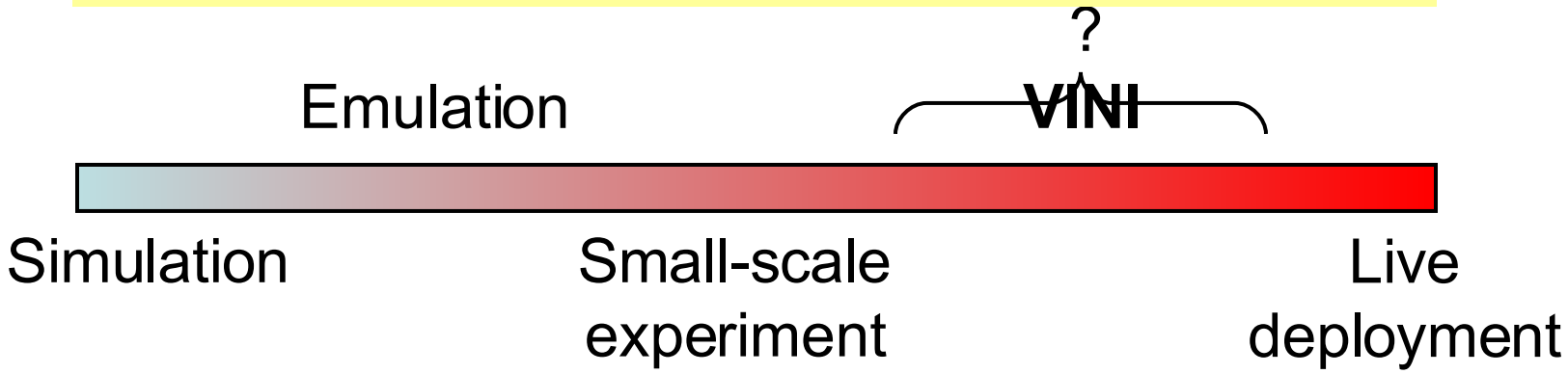
- Repeatability of network experiments is tricky
  - Proportional sharing
    - Minimum guarantees provided by limiting the number of outstanding shares
  - Work-conserving CPU scheduler means experiment could get *more* resources if there is less contention

PlanetLab for Network Architecture

- New components must be virtualized
  - Interfaces
  - Links
- Support for forwarding traffic over virtual links
- Stock and custom routing software

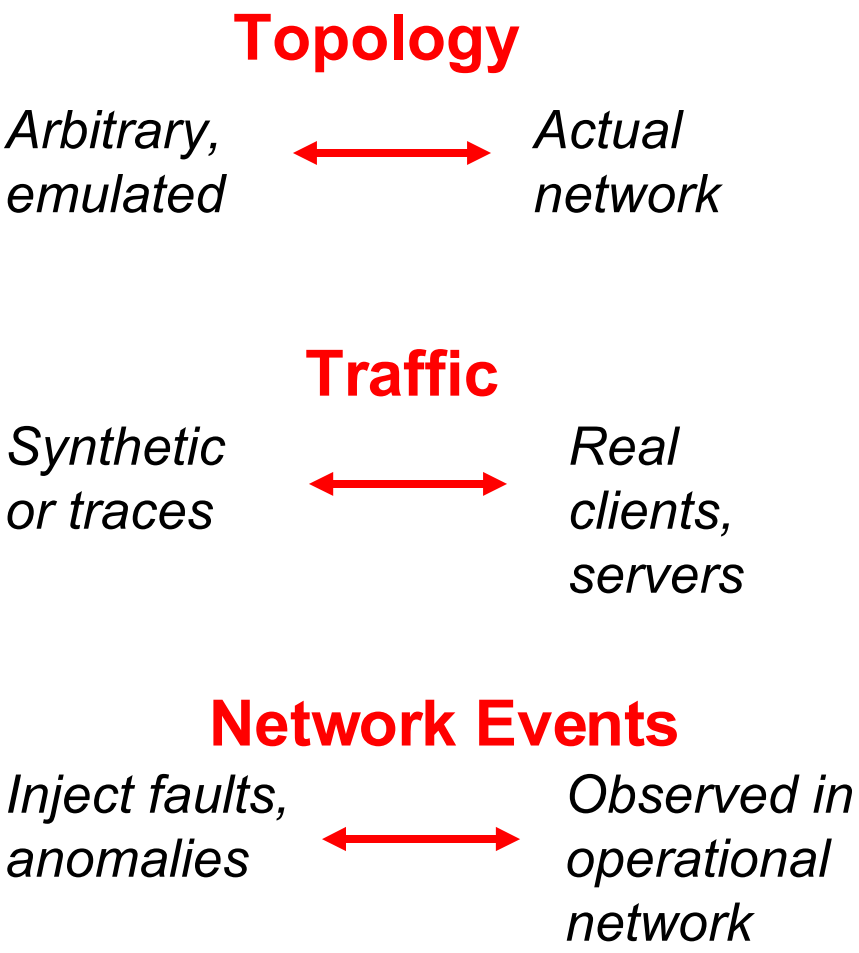
VINI Overview

Bridge the gap between "lab experiments" and live experiments at scale.



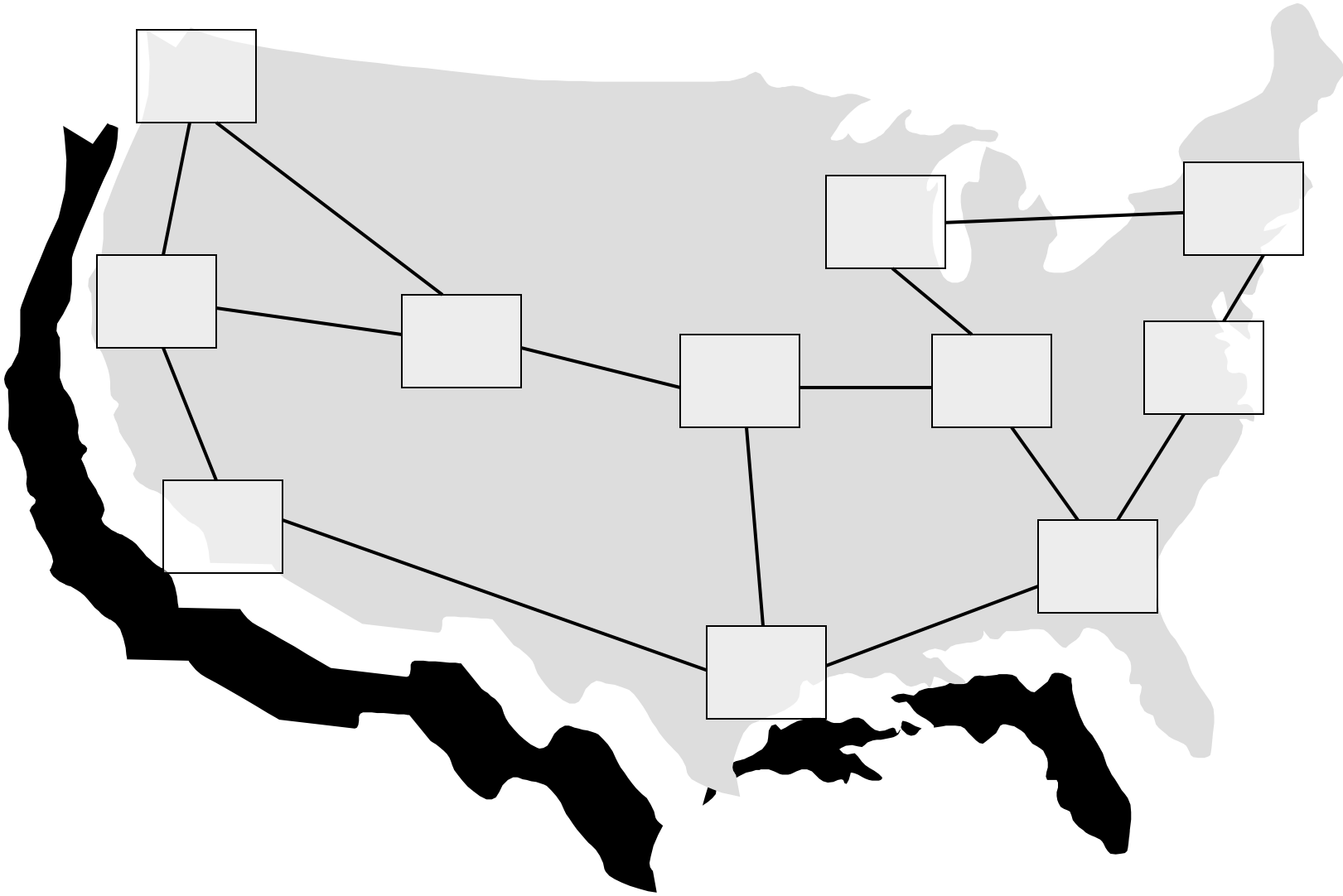
- Runs real routing software
- Exposes realistic network conditions
- Gives control over network events
- Carries traffic on behalf of real users
- Is shared among many experiments

Goal: Control and Realism

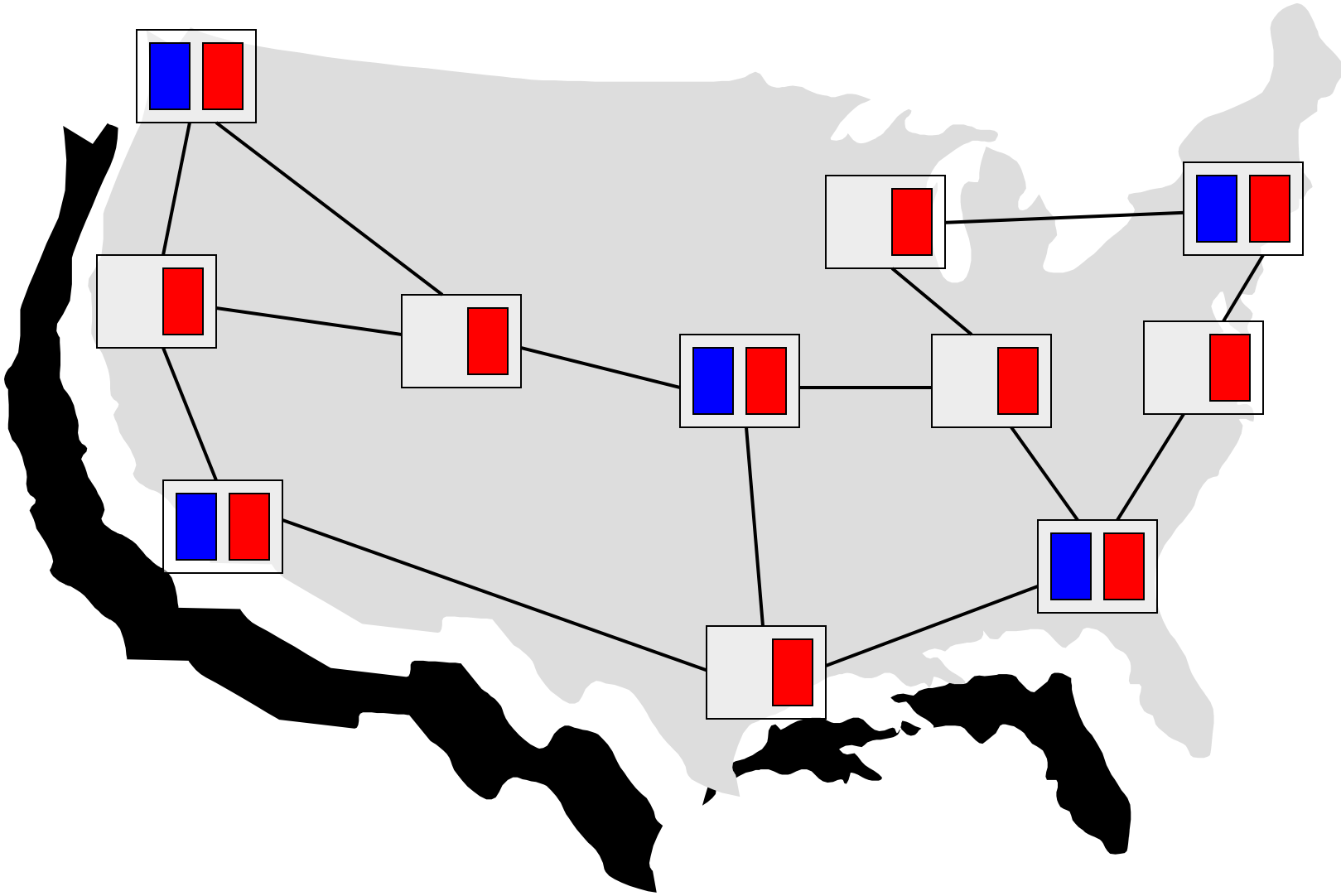


- **Control**
  - *Reproduce results*
  - Methodically change or relax constraints
- **Realism**
  - Long-running services attract real users
  - Connectivity to real Internet
  - Forward high traffic volumes (Gb/s)
  - Handle unexpected events

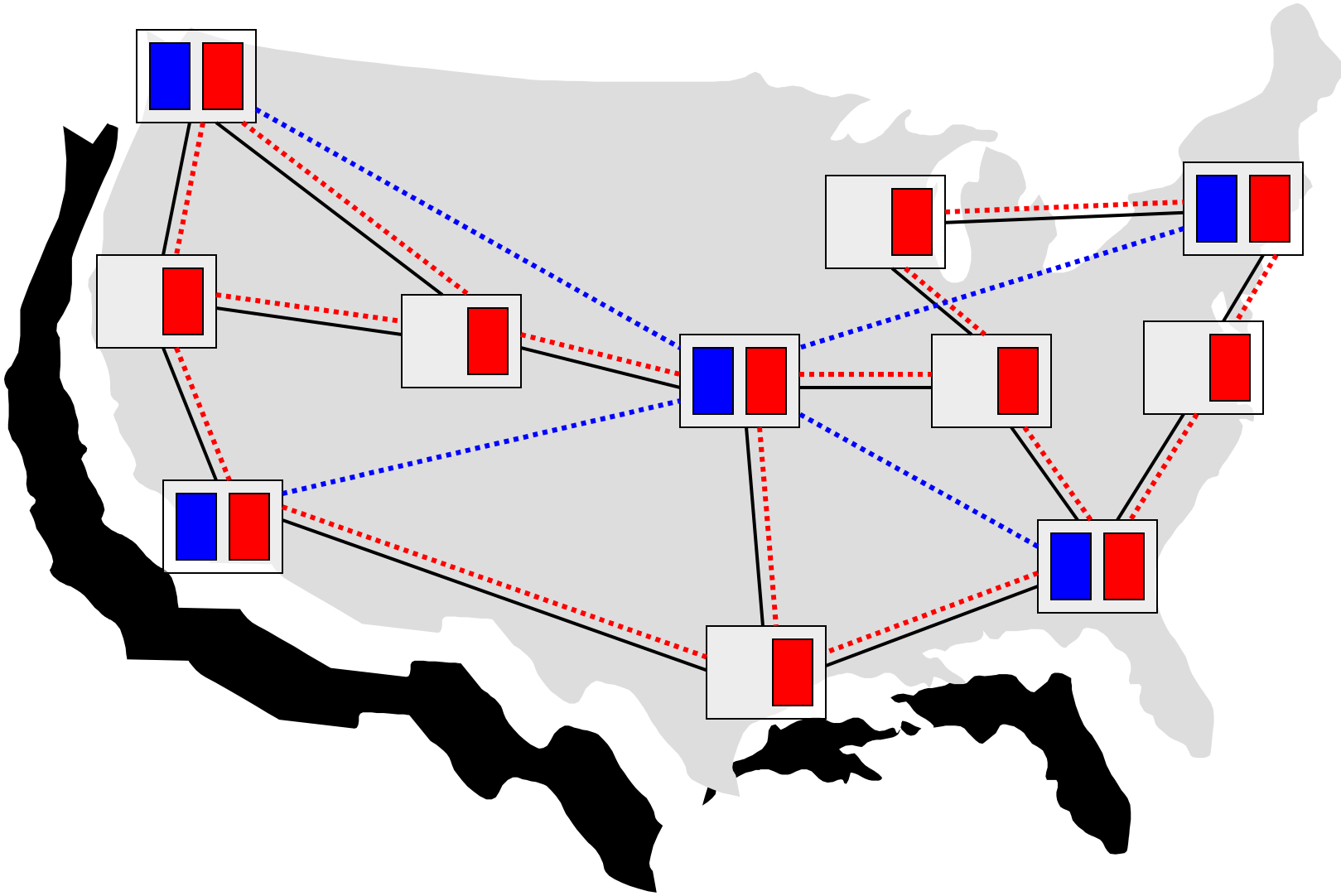
Fixed Physical Infrastructure



Shared By Many Parties



Supports Arbitrary Virtual Topologies



Why Is This Difficult?

- Creation of virtual nodes
  - Sharing of resources
  - Creating the appearance of multiple interfaces
  - Arbitrary software
- Creation of virtual links
  - Expose underlying failures of links
  - Controlled link failures
  - Arbitrary forwarding paradigms
- Embedding virtual topologies
  - Support for simultaneous virtual experiments
  - Must map onto available resources, account, etc.

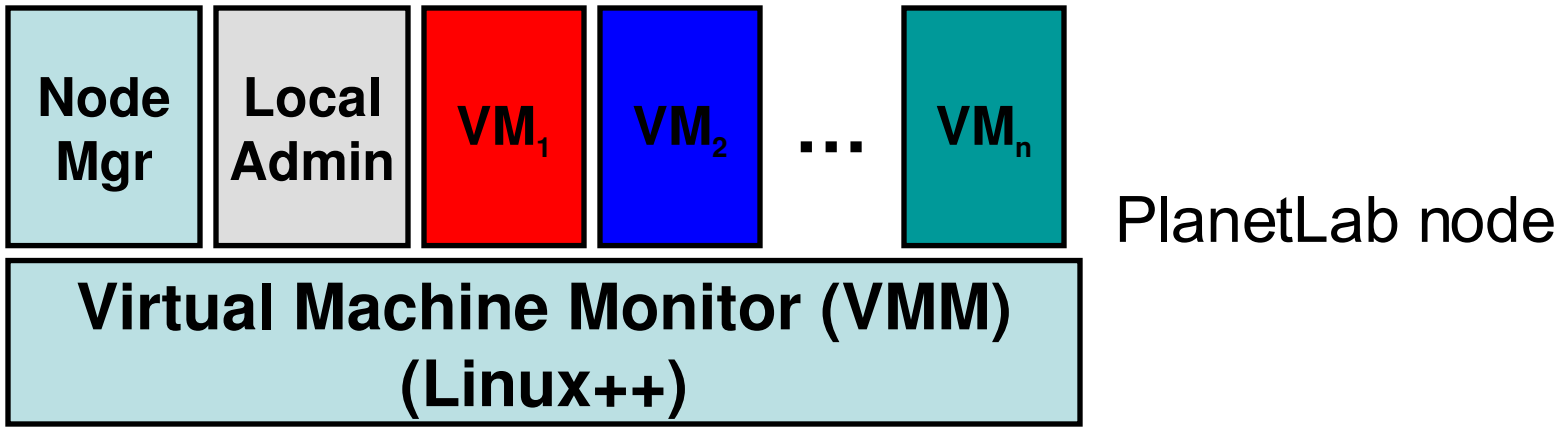
PL-VINI: Prototype on PlanetLab

- **First experiment:** *Internet In A Slice*
  - XORP open-source routing protocol suite
  - Click modular router

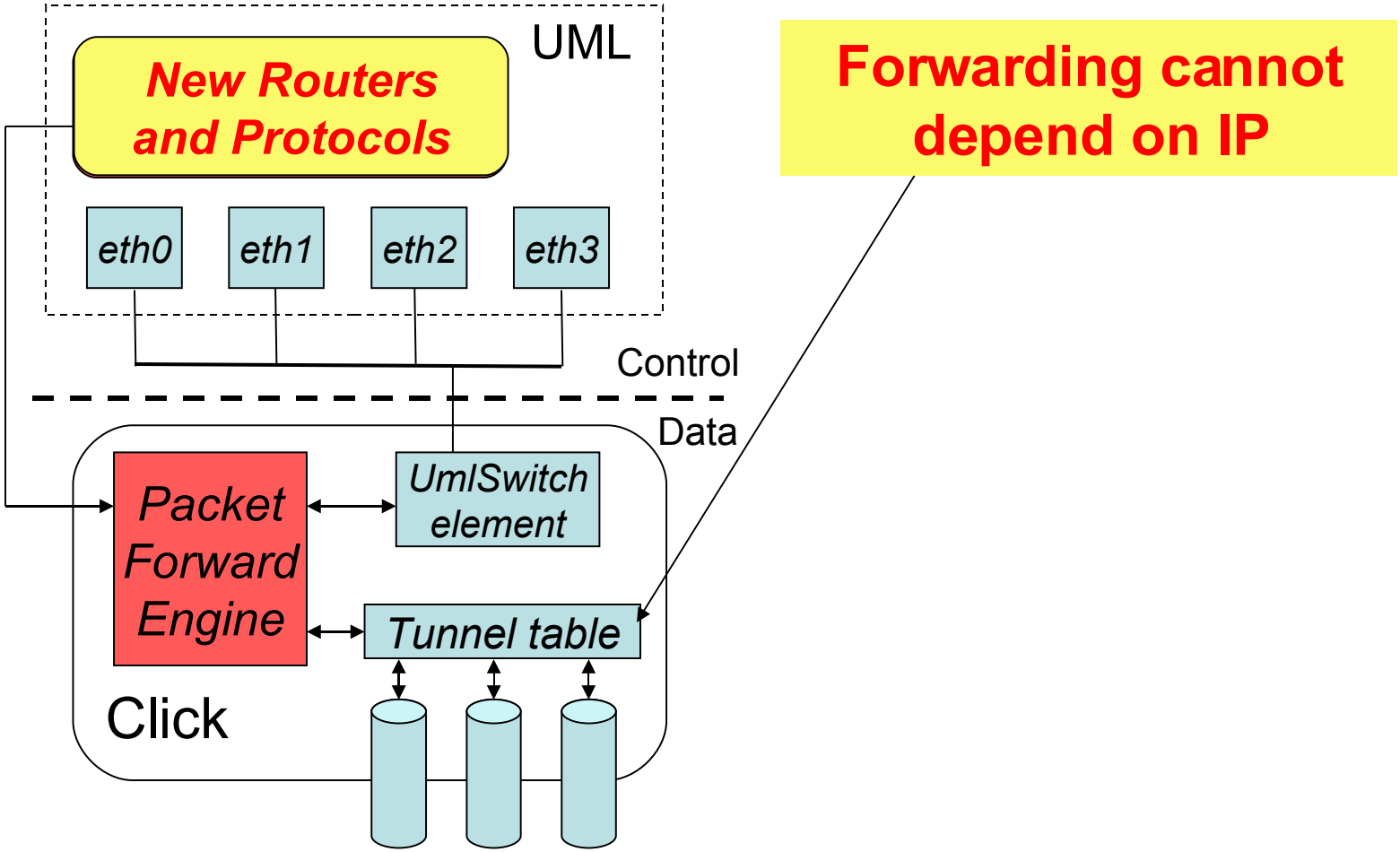
- Expose issues that VINI must address
  - Unmodified routing (and other) software on a virtual topology
  - Forwarding packets at line speed
  - Illusion of dedicated hardware
  - Injection of faults and other events

PL-VINI: Prototype on PlanetLab

- PlanetLab: testbed for planetary-scale services
- Simultaneous experiments in separate VMs
  - Each has “root” in its own VM, can customize
- Can reserve CPU, network capacity per VM



Recent Developments: Independence from IP



Demonstration