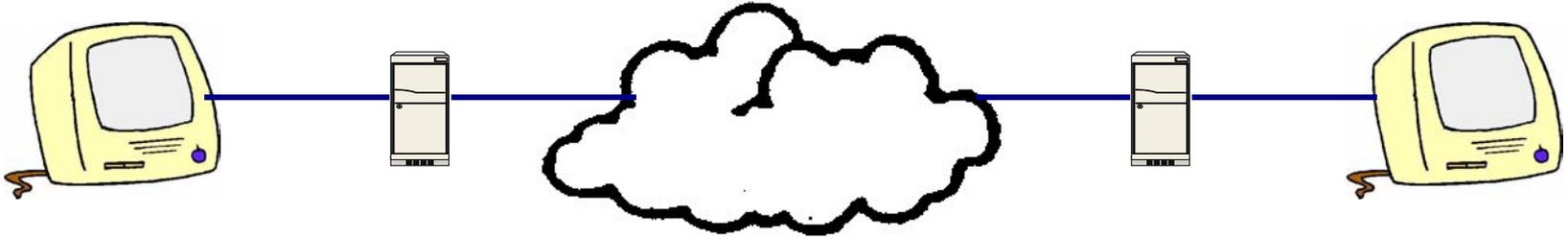


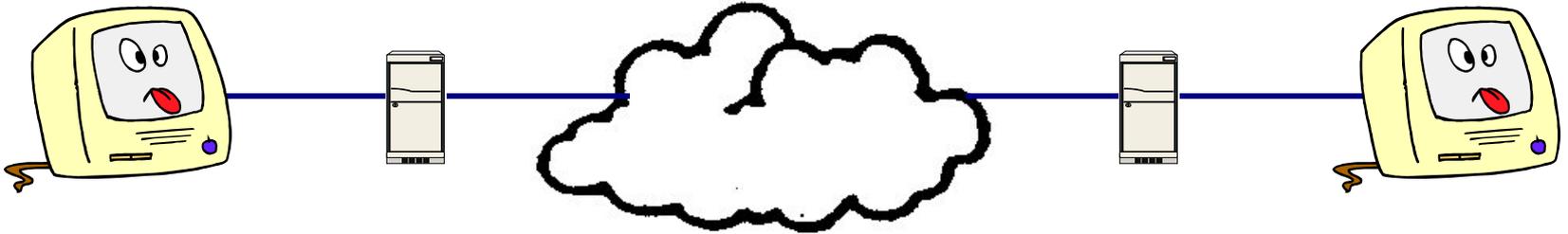
# Problem – Poor Performance



**You are experiencing poor network performance, but who is to blame?**

- Your End Host? Edge Router? Wide Area Network?
- A wide variety of network tools exist to help narrow down this problem
- However, most end-users are either:
  - Unfamiliar with these tools
  - Or, unable to fully understand the output of these tools

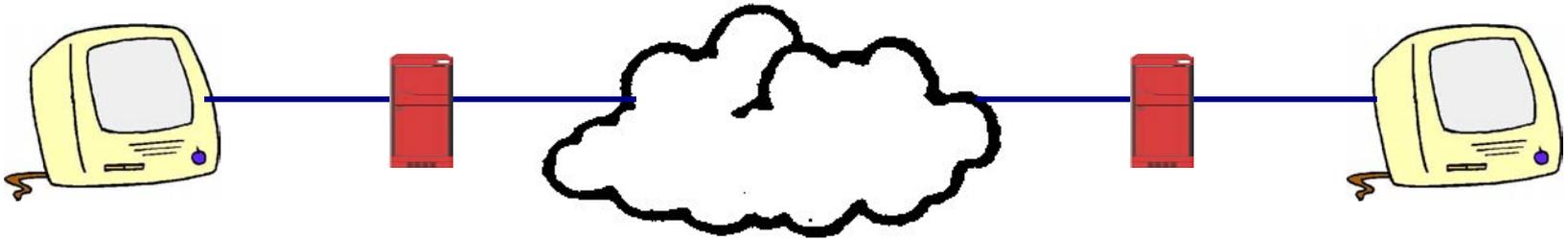
# Problem – At the End Hosts?



## End Hosts not tuned for optimal performance

- TCP uses a congestion window to determine how many packets may be sent at a given time
  - The larger the window, the higher the throughput
  - Maximum congestion window size is related to the space allocated for the send and receive buffers
- Must use optimal send and receive buffer sizes
  - Small buffers will cause the congestion window not to open fully
  - Large receiver buffer will cause the window to shutdown.

# Problem – At the Edge Router?



## Edge router could be misconfigured:

- Duplex mismatch is common
  - Edge router (or even the host itself) maybe set to half duplex instead of full duplex
  - Half duplex mode: Send or receive data, but not at the same time
  - Transfer rate is cut in half or more

# Problem – Wide Area Network?



## Problem could be in the Wide Area Network

- The path could be congested:
  - Packets are arriving at some router along the path much faster than it can forward them
    - Packets are dropped
    - Excessive TCP retransmissions are a sign of congestion
- There could be a bottleneck on the path:
  - Somewhere along the path is a much slower link
  - Causes the throughput to be significantly less than expected

# Solution: Performance Advisor

## Measures, displays, and analyzes network metrics

- Uses existing diagnostic tools:
  - ping, traceroute, Iperf, and Web100
  - integrates them into a common framework
- Attempts to emulate a junior-level network engineer:
  - Allows users to troubleshoot their own networking problems
  - Advises users on course of action, including “do nothing—your network performance is as good as possible”
- Additional tools and analyses are simple to add
- Implementation details:
  - Written in Java, uses XML-RPC for portability and extensibility

# Methodology: Architecture

## Performance Data Collector (PDC)

- Gathers network performance data

## Performance Data Historical Archiver (PDHA)

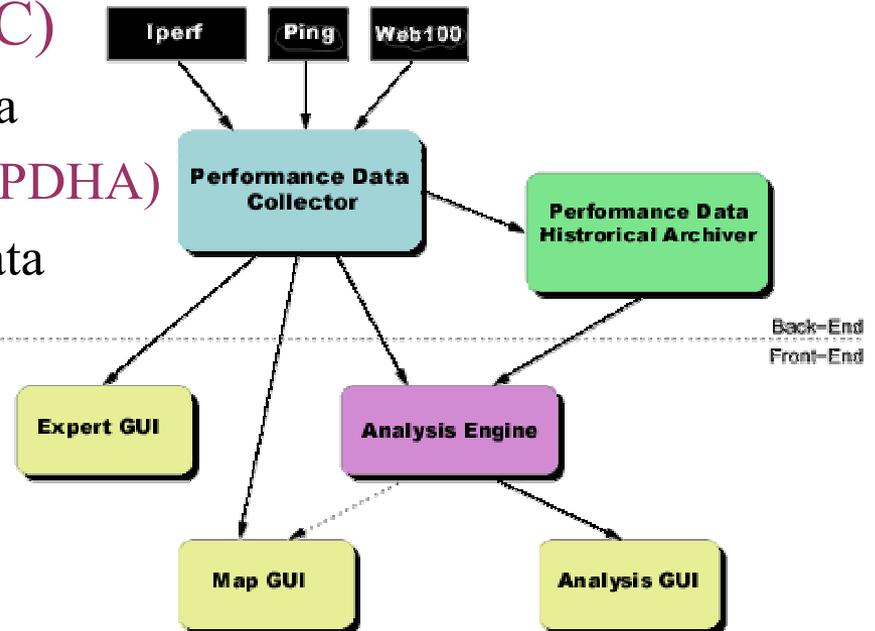
- Archives network performance data

## Analysis Engine

- Analyzes network data
- Provide plain text advice to solve problems or increase performance

## GUI

- Expert Interface: table & tree of metrics
- Map Interface: graphical display of network
- Analysis Interface: interact with Analysis Engine



# Performance Data Collector

- Designed to be stand-alone, extensible, and portable
- Elegantly handles platform differences and unavailability of any given measurement
- Features:
  - Uses *bundles* to facilitate integration of performance data measurement tools
  - Implements an XML-RPC interface
  - All requests are fulfilled immediately without any caching
  - Allows cooperation of both ends through a mechanism called activation (i.e. for tools such as Iperf)
  - Security using SSL and username/password (more to come)
  - Mechanism for autoupdating bundles

# PDC - Bundles

- A collection of scripts or Java classes that describe:
  - How to invoke a measurement tool
  - What metrics the measurement tool measures
  - How to parse the measurement tool's output
- PDC communicates with bundles using interface “Bundle”
  - canInvoke: returns true or false
    - Tests if bundle can be invoked (for platform specific tools)
  - getMeasures: returns metrics measured
  - getTool: returns tool it uses
  - getVersion: returns version of bundle
    - Used for auto-updating
  - needsActivation: returns true if activation on destination host is needed (for tools such as Iperf)

# PDC – Two Bundle Types

- A Java bundle can implement the Bundle interface
- An ADF bundle can consists of:
  - a set of executables which may be written in any language
  - a text file, called an *application definition file* (ADF), which includes the information needed to tie the pieces together
- Settings required in Application Definition Files:
  - measures: whitespace-delimited list of the metric identifiers
  - invoke: name of the bundle's executable
  - constraints: the name of the bundle's executable
  - tool: text that describes the method used to measure the metrics
  - version: version of bundle
  - activate: name of service this bundle provides
  - command: string containing activation executable name and arguments

# PDC - Features

- XML-RPC Interface
  - getAllMetrics: returns the list of metrics that may be measured
  - getMeasurement: returns an individual measurement
  - getMeasurements: returns a list of measurements
  - getAllMeasurements: returns all measurements given a remote host
- PDC auto-updating:
  - Periodically updates the bundles (automatically)
    - User can set how often to check for updates
    - All system bundles updated
  - Tool to update bundles on demand
    - User can control which bundles are updated

# PDHA (Design)

- **Short to medium-term storage of PDC measurements**
- **Features:**
  - Utilizes an XML-RPC interface
    - same security features as the PDC
  - Allow customization of how often, and how much historical performance data is stored
  - Act as a caching proxy for the PDC
  - Allow the retrieval of historical data by date and time:
    - Also from 3<sup>rd</sup> party databases
  - Allow different performance measurements to have different "lifetimes"

# Analysis Engine

- Analyze the metrics for a specific end-to-end path and give advice to solve any performance or connectivity problems
- Features:
  - “Test Definition Files” (TDFs) similar to PDC’s ADFs
    - TDFs consist of detection rules (simple binary operations on metrics), problem descriptions, and solutions.
    - Will migrate to an interface similar to the *bundles*
  - Constructs decision trees to efficiently determine problems, driven by the detection rules
  - Decision trees are ordered by cost and probability
    - Cost is updated by results from the PDC

# Example: ADF Bundle

## **iperf.adf:**

Measures: network.link.bandwidth.available.TCP

Invoke: invoke.sh

Constraints: constraints.sh

Tool: iperf

Version: 2.3+5.0

Activate: iperf-server

Command: /usr/bin/iperf -s

---

## **activate.sh:**

```
#!/bin/sh
```

```
#activate the iperf server
```

```
iperf -s & echo $!
```

# Example: Java Bundle

```
public class TestJava implements Bundle {
    static final private String METRIC =
        "test.java.measurement";
    static final private String TOOL = "testing-
        java";
    static final private String VERSION =
        "1.2.3+3.1.2";

    public boolean canInvoke() {
        return true; }

    public MetricList getMeasures() {
        MetricList ml = new MetricList();
        ml.add(METRIC);
        return ml; }

    public String getTool() {
        return TOOL; }

    public String getVersion() {
        return VERSION; }

    public boolean needsActivation() {
        return false; }

    public boolean activationServiceVerified() {
        return true; }
```

```
public String getActivationService() {
    return null;
}

public String getActivateCommand() {
    return null;
}

public MeasurementMap invoke(InetAddress
    remoteHost, int port)
    throws AdvisorException {
    try {
        MeasurementMap mm = new MeasurementMap();
        Measurement m = new Measurement(METRIC,
            "2", TOOL, InetAddress.getLocalHost(),
            remoteHost, new Date(), new Date()); mm.put(METRIC,
            m);

        return mm;
    } catch (UnknownHostException ex) {
        throw new AdvisorException(ex); }
}
}
```

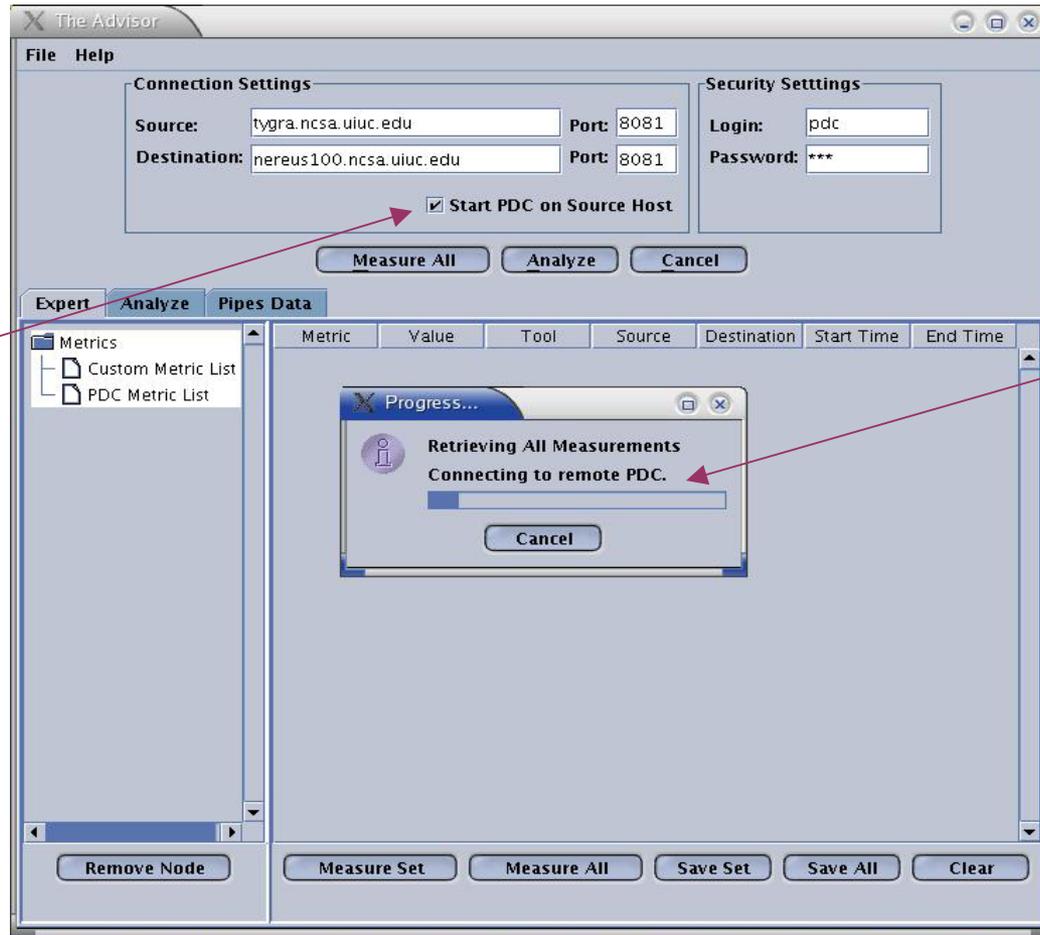
# For More Information

- Visit our website:
  - <http://dast.nlanr.net/Projects/advisor>
- Download alpha release
  - PDC and Expert GUI only
  - Released: November 2003
- Join our mailing list: advisor-users
  - Instructions on website

# Advisor GUI

- Three main graphical displays
  - Expert GUI: Displays all metrics
    - Uses a tree to organize metrics
    - Uses a table to display metrics and corresponding information
  - Analysis GUI: Displays advice reported from the analysis engine
    - Simple text display
  - Map GUI: Visually display of the network and trouble spots
    - Users will be able to click on the map to view measurements for specific areas along the path

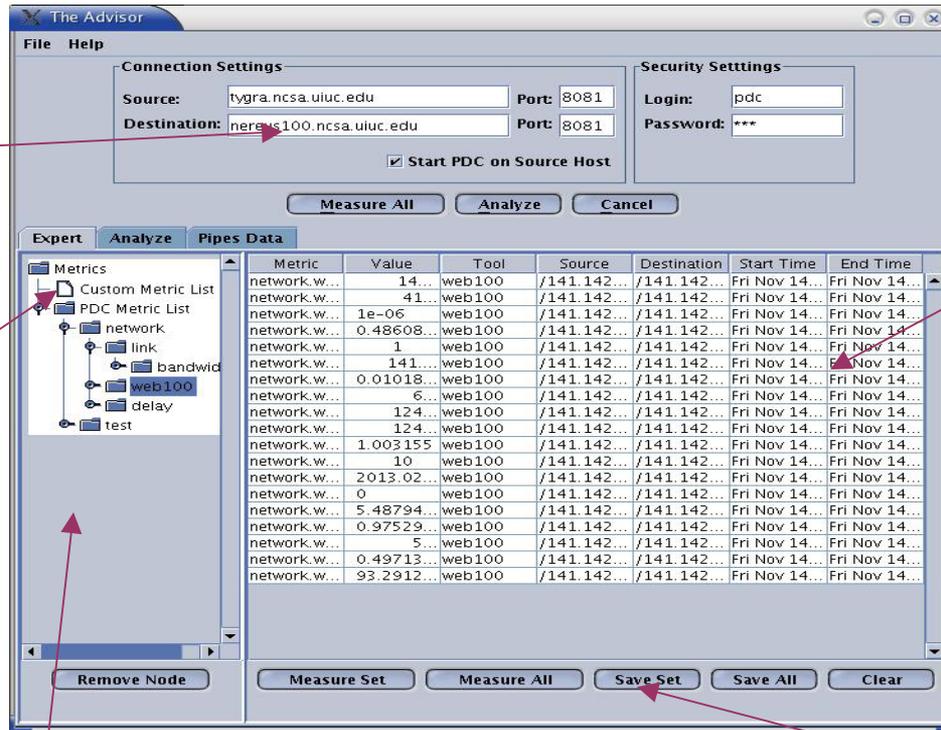
# Advisor Expert GUI



**Optionally  
start PDC on  
local host  
automatically**

**Progress  
Monitor**

# Advisor Expert GUI



*Connection and security settings*

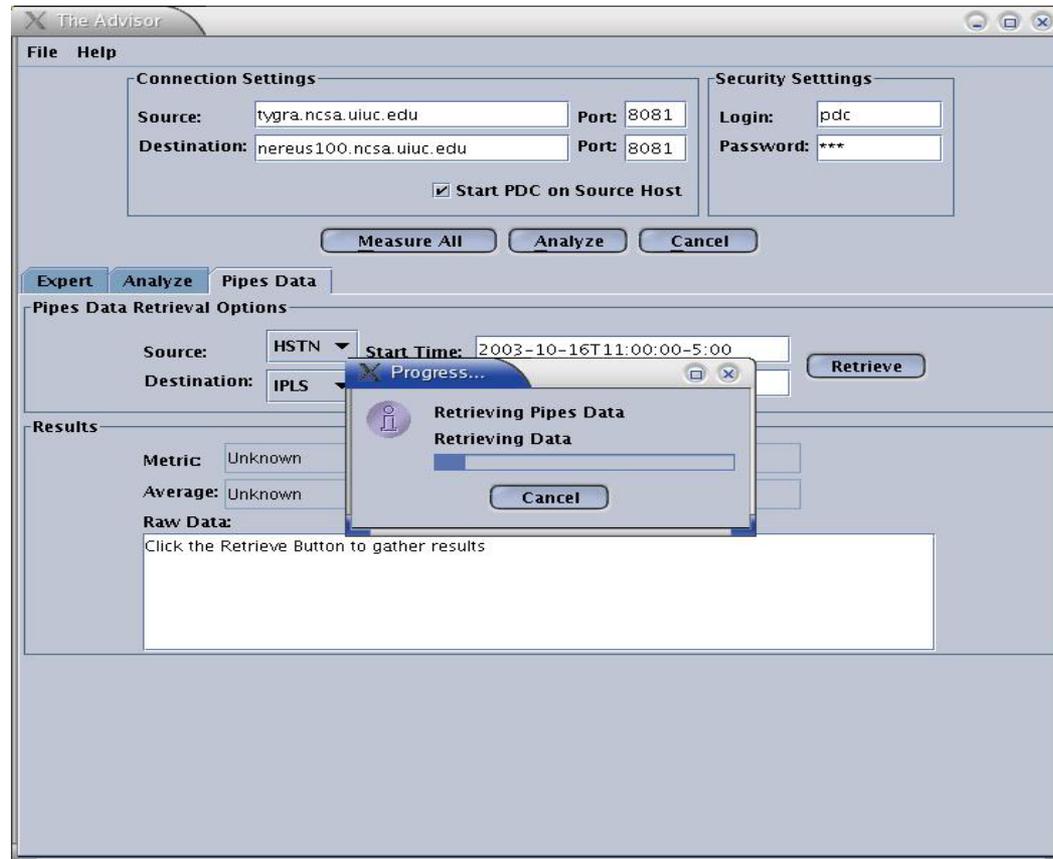
*Table showing metrics selected in the tree display*

*Custom metric list to watch specific metrics or constrain the list*

*Tree display of all metrics*

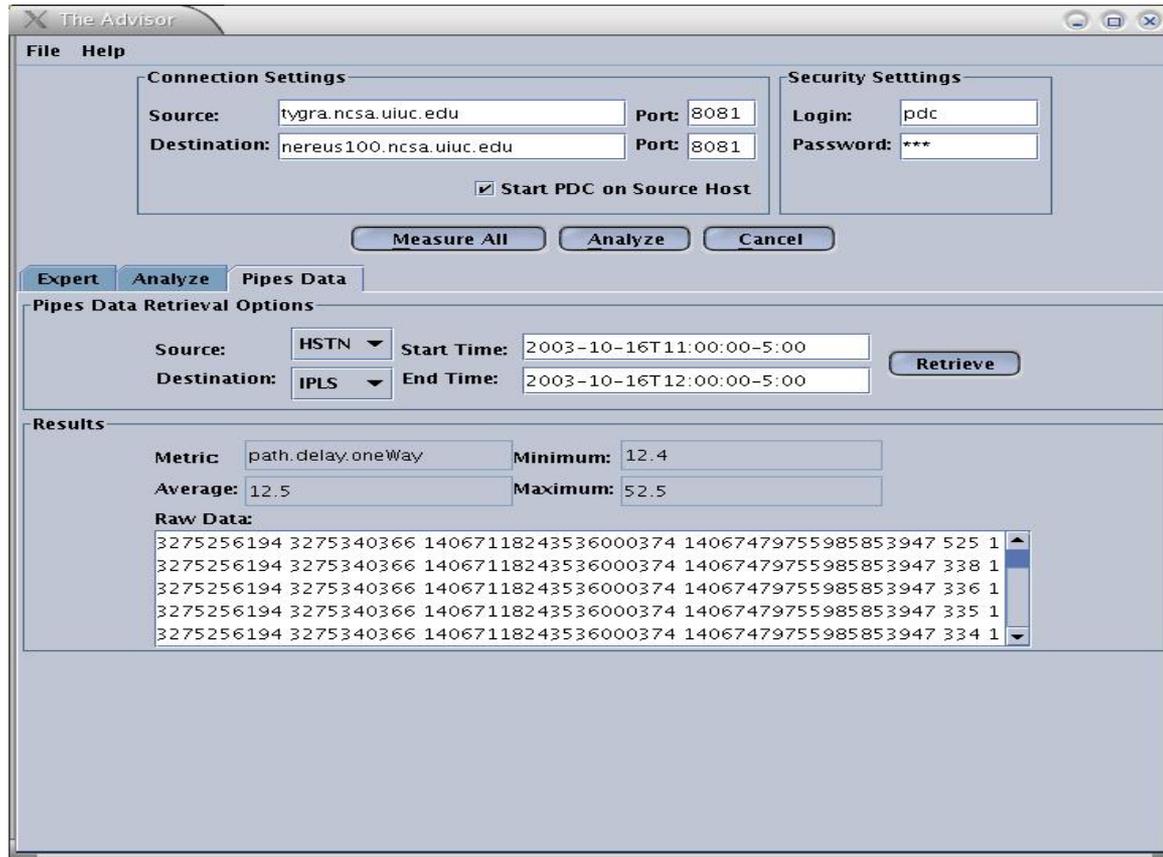
*Buttons to measure the current metric set in the table, all metrics, save the current set, save all metrics, and clear the table.*

# Advisor GUI



- Retrieving Internet2 Pipes Data
  - [http://e2epi.internet2.edu/E2EpiPEs/e2epipe\\_index.html](http://e2epi.internet2.edu/E2EpiPEs/e2epipe_index.html)

# Advisor GUI



- **Displaying Internet2 Pipes Data**

- [http://e2epi.internet2.edu/E2EpiPEs/e2epipe\\_index.html](http://e2epi.internet2.edu/E2EpiPEs/e2epipe_index.html)

# Poster Credits

- **Tanya Brethour, NLANR/DAST**

Email: [brethour@ncsa.uiuc.edu](mailto:brethour@ncsa.uiuc.edu)

- **Jim Ferguson, NLANR/DAST**

Email: [ferguson@ncsa.uiuc.edu](mailto:ferguson@ncsa.uiuc.edu)