

Resource Monitor Design

Resource Monitor is responsible for gathering information about resources that affect communications performance, such as: available system bus bandwidth, available bandwidth on the network and CPU utilization. It can also perform prediction based on the available information and scheduler request. The goal of the Resource Monitor is to provide accurate measurements of the performance of specific resources and effective statistical forecasting based on time-series analysis. It'll be a robust distributed system that can perform active and passive monitoring function as well as prediction dynamically for both copper wire network and optical network.

1 Resource Monitor Architecture

Resource Monitor consists of four components:

- Monitor

There are two types of Monitors: NetworkMonitor and HostMonitor. NetworkMonitor is responsible for collecting network information (see details in “Metrics For Network Measurement”). HostMonitor is responsible for collecting host information (see details in “Metrics For Host Measurement”).

- Resource DB

Store the information gathered by Monitor.

- Predictor

Apply time-series models to the data collected in the Resource DB to predict the next value in the series. Predictor have two parts: NetworkPredictor and HostPredictor.

- Data Analyzer & Presentation

Data Analyzer analyze the collected information from Resource DB periodically/real time and store the results back to the Resource DB. User can view the monitoring and forecasting information by Web presentation tool.

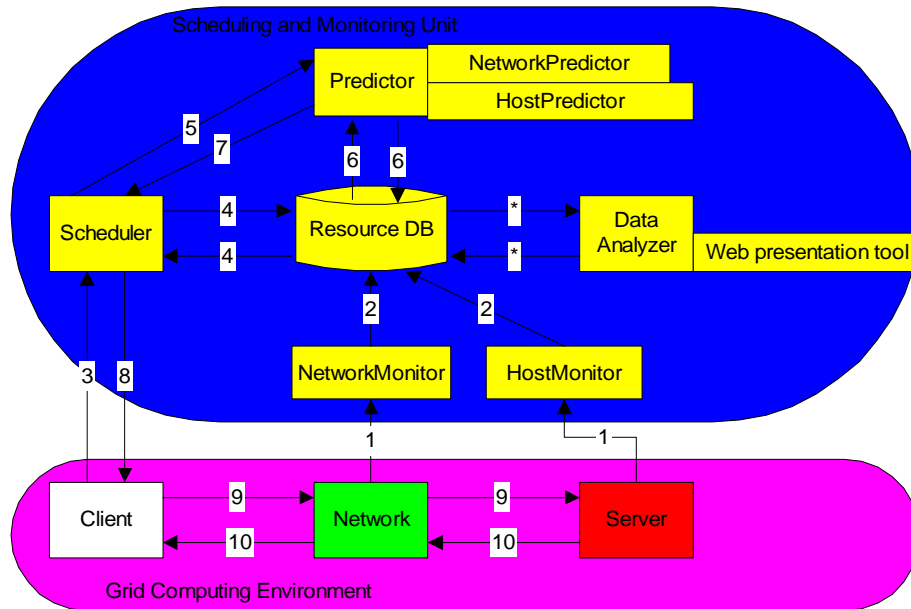


Figure 1 Resource Monitor Architecture

Figure 1 shows the Resource Monitor works in the following steps:

1. NetworkMonitor monitors network periodically/real time; HostMonitor monitors server periodically/real time.
2. NetworkMonitor and HostMonitor store observed information into ResouceDB real time.
3. Client invokes task and talks to scheduler to inquire suitable server.
4. Scheduler queries available servers from ResouceDB and gets feedback information.
5. Scheduler queries prediction from Predictor.
6. Predictor performs prediction by querying information from ResourceDB.
7. Predictor gives prediction result to scheduler.
8. Scheduler performs scheduling and returns scheduling information to Client.
9. Client sends task to Server through Network.
10. Server execute task and returns result to Client through Network.

There are three categories of NetworkMonitor: end-to-end active monitor, end-to-end passive monitor, route trace monitor (Shown in Figure 2).

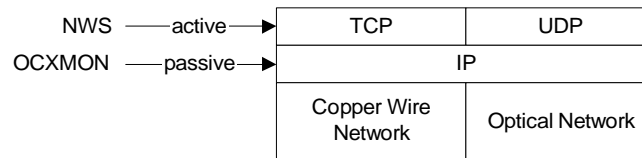


Figure 2 NetworkMonitor Categories

The end-to-end active network monitor probes network resources by conducting end-to-end network probes and recording the resulting performance. To conduct a probe, one host opens a connection with another and sends a small message to measure the link round-trip time, and a large one to measure the throughput. The Network Weather Service (NWS) is an active monitoring system which uses end-to-end TCP/IP probes to measure network performance. Since it can provide accurate measurements of TCP/IP traffic and some possible effective statistical forecasting, we could integrate NWS into Resource Monitor as the end-to-end active monitor and NetworkPredictor for TCP/IP traffic.

The end-to-end passive network monitor does non-invasive network resource measurement to the observed networking environment. OCXmon monitor is a good example of this. It taps into the light of a fiber interconnection by means of optical splitters, and collect packet header traces. The abstraction of packet header traces can generate an immense amount of data and can happen at a central data collection location or at the location where the data is being collected in real-time or non real-time. Since the OCXmon monitor usually connect to network by its ATM card to monitor IP traffic, we can integrate OCXmon monitor into Resource Monitor as the end-to-end active monitor and DataAnalyzer for IP traffic.

The pathchar is a good example for route trace monitor which collect route information between specified end points. The integration of it will be based on the actual requirement.

Integration of NWS and OCXmon will be the first step for implementation of end-to-end active monitor and passive monitor for TCP/IP traffic. To integrate other monitoring software or to implement active monitor and passive monitor for UDP or RBUDP will be the next step based on the actual requirement.

2 Hierarchy of Resource Monitor

Instead of having all NetworkMonitor of a system asynchronously conducting experiments with each other, the administrator organizes the system as a hierarchy of hosts sets called cliques. A clique consists of a set of hosts where a given network probe is conducted by each machine with other members of the clique with a given period. Figure 3 shows a system that has been set up on fifteen machines. The three domains that are monitored include five hosts. The probe conducting sequence in each clique will be based on some kind of protocols, such as NWS token protocol, to avoid measurement intrusiveness or confliction.

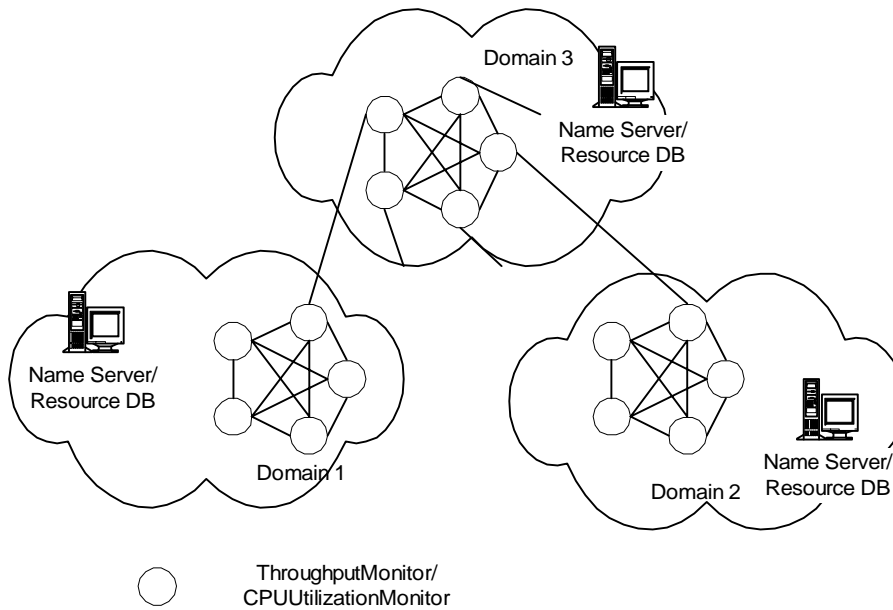


Figure 3 Hierarchy of Resource Monitor

3 Metrics for Network Monitoring

- **Packet Loss:** provides a good measure of the quality of the route between end points. However the manner in which applications can deal with such loss will vary greatly depending upon their use of TCP or UDP and the particular requirements of the application.
- **Round Trip Time (RTT):** is the time taken to traverse the path from the source to the destination and back. Formally, given a packet p , the time at which the last packet byte departs from the source $t(D)$, and the time at which the last packet byte arrives at the packet destination $t(A)$, $RTT = t(A) - t(D)$. The RTT is the sum of the propagation time between the end points plus the queuing delays introduced at each hop along the path between the sites. It is therefore characterized on the distance between the end points of the route, the number of router hops on the route and the delay encountered at each hop. It is a "there and back" measurement which has the benefit that timing measurements are confined to the source, i.e. there is no need for clock synchrony between source and destination. On this basis it is a simple metric.
- **One-Way Delay:** measures the path between source and destination and is the sum of the propagation delays of the data links and the delay introduced at each router hop on the path. One-way delay measurement requires external clock sources (like GPS or NTP - depending on the precision required) for synchronization and the co-ordination of source and destination processing to make the measurements. One-way delay measurement has the benefit of providing the measurement of a specific path through the Internet and recognizes that asymmetric routing commonly occurs within the Internet. However for an application, the communication between it and the remote client is what matters and regardless of the particular routes taken for the traffic, it is the "there and back" characteristics that matter.
- **RTT / One-way Delay relative to a Given Route:** is dependent upon the route taken across the network and such route variations are likely to introduce differing transit delays. For example, a provider may choose to

route traffic on a particular link because of capacity and take no account of distance and associated delay; or because of fault conditions route flapping will have an indeterminate effect of transit delay.

- **Variation in RTT (frequency distribution of RTT):** A plot of the frequency distribution of rtt measurements can be used to provide a reasonable estimation (the inter-quartile range) of jitter. The benefit of this approach is that variation in RTT can be readily computed using the data gathered is the simple measurement of RTT.
- **Jitter:** the variation in arrival times of successive packet from a source to a destination. It is formally defined by the IETF as the "instantaneous packet delay variation" (IPDV) and is the difference experienced by subsequent packets, I and I+1, on a one-way transit from source to destination. The measurement of one-way delay and derived IPDV provides a means whereby a more rigorous characterization of the Internet can be developed.
- **Volume (number of (Grid) bytes exchanged):** the measure of the total number of bytes exchanged per specified Grid activity. Traffic volume estimated for each transaction on a daily/weekly/monthly basis would provide value.
- **Per Flow Application Throughput:** defines the throughput (byte/s) measured for a specific Grid application between specified end-points, i.e. DA/SA/Port. Typically it will be based on the measurement of the actual data transferred during the exchange, i.e. a "passive" measurement, and not on additional (test) data inserted into the network. However the capability for "active" measurement of throughput will be required through the injection of test traffic using applications like NWS. This will allow knowledge of network capability to be recorded and problems identified ahead of use by real GRID application traffic.
- **Aggregate Network Throughput:** defines the aggregated throughput within the network between source and destination end points. This may

measure the current utilization as a rate (volume/time) or as a proportion of the total capability within the path across the network.

4 Metrics for Host Monitoring

- static host information (operating system version, CPU type, number of processors, etc.).
- dynamic host information (load average, queue entries, etc.).
- storage system information (available disk space, total disk space, available memory etc.).

5 Resource DB

Resource DB can store information as specified format files or into LDAP server. Grid Application can access Resource DB via LDAP Schema for monitoring metrics (shown in Figure 4).

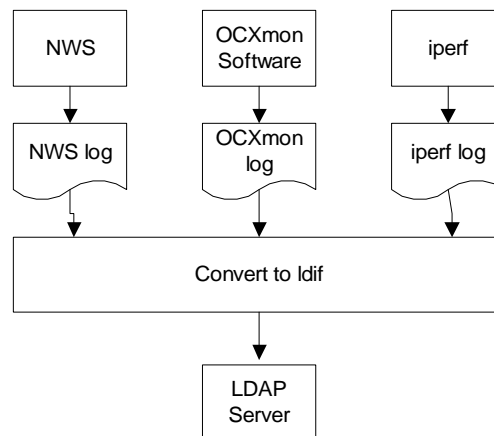


Figure 4 Store to LDAP server

6 NWS Architecture

The NWS is also a modular system which is built from different component processes that run on the hosts the user wants it to monitor. According to its needs, the administrator can dynamically remove or add components to its system. Four components are available:

- Name Server

manages the correspondence between the IP/domain address for each independently-running modules of NWS. It records the set of time series that are actively being gathered by the system and the host locations where they are stored.

- Memory Server (→ResourceDB)

Measurements that have been taken are then remotely stored as time stamp-measurements pairs in memory hosts. The data is immediately written in a file so that it can survive a failure of the process.

- Sensor (→NetworkMonitor/HostMonitor)

A sensor periodically takes measurements of the available performance of some resource and sends it to a memory component. Sensors can be dynamically reconfigured through specific messages sent by the user.

- Forecaster (→Predictor)

When a forecaster is asked for a prediction, it contacts the name server to learn the location of the data, downloads it from the specified memory host, applies time-series models to the data received, and delivers the forecast to the scheduler.

Using NWS C-based API or Java API, we can integrate NWS into Resource Monitor (shown in Figure 5). Currently, the latest version of NWS can provide accurate measurement of CPU availability, disk space availability, memory availability, end-to-end bandwidth, latency and connection time on TCP/IP network.

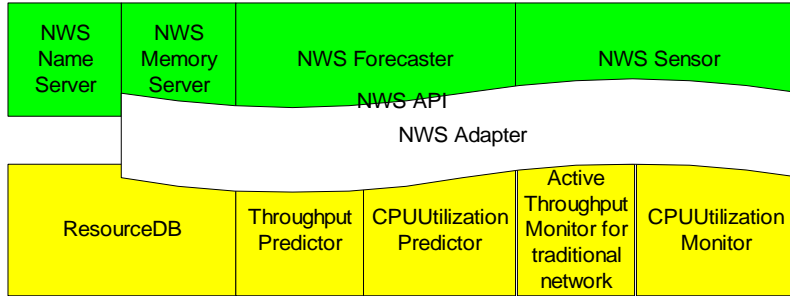


Figure 5 Integrate NWS into Resource Monitor