

# Topology Discovery Services for Monitoring the Global Grid

*Luca Valcarengi, Luca Foschini, Francesco Paolucci, and Piero Castoldi, Scuola Superiore Sant'Anna  
Filippo Cugini, CNIT National Lab of Photonic Networks*

## ABSTRACT

The dynamic joint optimization of both computational and network resources has the potential of guaranteeing optimal performance to geographically distributed grid applications. A grid Network Information and Monitoring Service (NIMS) has been recently proposed to complement computational resource status information with network resource status information. NIMS information includes, but is not limited to, information already available in the network control plane (e.g., network topology, link capacity occupation, communication delay).

This study first reviews some measurement methodologies and network sensors suitable for implementing NIMS components, and then describes some tools currently utilized for monitoring grid network infrastructures. Finally, two implementations of a NIMS component, called the Topology Discovery Service (TDS), are proposed and evaluated. The TDS provides grid users (e.g., applications) or the programming environment middleware with up-to-date information on the grid network infrastructure topology and status. Both proposed implementations can be utilized in any global grid network based on commercial routers without requiring modifications of router management and control protocols.

## INTRODUCTION

The increased capacity of networks connecting geographically distributed computational and storage resources is pushing grid computing-enabled applications from local area networks (LANs) (i.e., a local or cluster grid) to wide area networks (WANs) (i.e., the global grid). However, while cluster grid applications may generally exploit almost-dedicated network resources, in the global grid network resources are shared among heterogeneous applications. Thus, to guarantee the required quality of service (QoS) to global grid applications, the need for services providing dynamic and deterministic network resource reservation is rapidly emerging. Grid Network Services [1, 2] have been introduced to fulfill this need: they allow either the application

or the grid programming environment to dynamically adjust the utilized network resources for seamlessly meeting the application QoS requirements. Grid Network Services include, but are not limited to, network information, monitoring, connectivity, and cost-estimation services.

In this study the Network Information and Monitoring Service (NIMS) is considered [1, 3]. The NIMS consists of a set of services that provide network resource status information. This study first reviews methodologies and network sensors currently available for measuring network performance/status. Then it summarizes the various ongoing efforts in the grid community for implementing NIMS components. Finally it proposes and evaluates two implementations of a NIMS component, namely, the Topology Discovery Service (TDS). The TDS is a grid Network Service that, similarly to the Network Capability Discovery Service [2], provides either the grid users or the grid middleware with a snapshot of the current grid network's infrastructure status. The status is defined through different traffic engineering (TE) parameters, such as node adjacency at different network layers, bandwidth, and latency.

The two proposed TDS implementations are based on a consumer/producer architecture [3] and are designed to operate in networks based on IP/MPLS commercial routers connecting distributed computational resources. The main differences between the two implementations consist in the producer and consumer functionality assignment, the methods utilized for obtaining network status information, and the type of retrieved information. The centralized implementation, centralized TDS (C-TDS), provides information related to both the physical and the Multi-Protocol Label Switching (MPLS)-layer topology by accessing router information databases. The distributed implementation, distributed TDS (D-TDS), allows users to discover the links along which IP packets are routed (i.e., as defined in this study, the network logical topology) and some of their performance metrics (e.g., link capacity and communication latency) by utilizing network sensors available to generic grid users. On the one hand, the C-TDS is able to obtain very detailed information about

the network status, but this requires accessing each router internal database. On the other hand, the D-TDS trades the level of details of the retrieved logical topology information with the topology discovery scalability.

## THE NEED FOR GRID NETWORK SERVICES

In distributed grid applications, the dynamic reallocation of tasks based on the utilized computational and network resources allows optimization of application performance [4]. Within the Grid High-Performance Networking Research Group (GHPN-RG) of the Global Grid Forum (GGF), the implementation of Grid Network Services has been proposed to optimally perform reallocation of both computational and network resources dynamically and transparently to the user [1, 2]. In particular, within the GGF, the Network Measurement Working Group (NM-WG) defines performance metrics, measurement methodologies, and network sensors commonly utilized for implementing grid monitoring tools. In the following subsections, the study done by the NM-WG and the ones presented in [5, 6] are summarized. Furthermore, the current implementations of grid monitoring tools that can be part of the NIMS are outlined with their advantages and drawbacks.

### NETWORK PERFORMANCE METRICS

Sample network performance metrics of interest are the following:

- *Bandwidth capacity* is the maximum amount of data per time unit of either a point-to-point link (at a specific network layer) or end-to-end-path.
- *Bandwidth utilization* is the aggregate capacity currently utilized on a link or a path.
- *Available bandwidth* is the difference between the bandwidth capacity and the bandwidth utilization over a given time interval.
- *Achievable bandwidth* is the maximum amount of data per time unit of either a link or a path given some protocol and hardware specifications, such as transmission protocol (e.g., TCP or UDP) and operating system settings (e.g., TCP buffer size).
- *One-way delay (OWD)*, is the time required for a packet to travel from source to destination.
- *Round-trip time (RTT)*, is the time required for a packet to travel forth and back between a source and a destination.
- *Loss* is the fraction of packets lost during a communication session between a source node and a destination node.

### MEASUREMENT METHODOLOGIES

In this section, some of the currently utilized measurement methodologies are reviewed.

In the *packet-pair* method, the minimum bandwidth capacity (i.e., the bandwidth capacity of the bottleneck link) in an end-to-end path is calculated by sending consecutive pairs of packets. If noncompetitive traffic is present along the

path, packets are queued on the slowest (bottleneck) link. If no pipeline mechanism is present, a gap between the packets appears after the bottleneck link because of its low transmission rate (i.e., low capacity). The time gap is estimated over a set of pair probes so that the bandwidth capacity of the slowest link is computed as follows:

$$C = P/gap, \quad (1)$$

where  $P$  is the packet size in bits and  $gap$  is the temporal spacing between packets in seconds.

The *packet-train* method generalizes the packet-pair method by sending a homogeneous packet set instead of packet pairs. The interarrival times are evaluated between pairs belonging to the same set.

The variable packet size (VPS) method allows estimation of each link bandwidth capacity. Packets of different size are sent through the network with different values of the TTL field. After traversing a known number of links, determined by the IP TTL field, the packets are discarded and an ICMP message is sent back to the sender. In this way the sender is able to compute the RTT. The RTT values obtained for each node are linearly interpolated, so that the difference between the slope of the RTT of two consecutive nodes is the inverse of the bandwidth of the link which connects the two nodes. VPS-like methods can be improved by the utilization of the even/odd mathematical tool.

The *tailgating* method (used, for instance, by the *nettimer* tool [7]) attempts to further improve the packet-pair method by utilizing a two-phase approach. During the first phase (i.e.,  $\sigma$  phase) the performance along an end-to-end path is measured using a VPS-like technique tuned for end-to-end measurement; during the second one (i.e., tailgating phase) link-based measures are taken. Two packets are sent in the latter phase. The first has the same dimensions of the link MTU and has TTL equal to the distance of the link under measurement. The second packet has the minimum size allowed by the protocol. As long as the TTL of the first packet is not expired, the second (smaller) packet follows the first. When the TTL of the first packet expires, the second one proceeds without being queued to the first one anymore. This mechanism is iterated for each link until the destination is reached. In this way, it is possible to estimate the bandwidth of intermediate links by comparing the delay accumulated by the smaller packet with the (fixed) delay of bigger packets. Compared to a plain-packet pair, packet tailgating usually consumes less network bandwidth, and does not rely on consistent behavior of routers handling ICMP packets and on timely delivery of acknowledgments. However, the measurement accuracy does not usually outperform the packet-pair method for path longer than a few hops.

The self-loading periodic streams (SLOPS) method measures the available bandwidth over an end-to-end path by sending many packet streams at a rate greater than the supposed path available bandwidth. The packets are therefore queued at the node to which the smallest available bandwidth link is connected. Each packet is

*In the packet-pair method, the minimum bandwidth capacity (i.e., the bandwidth capacity of the bottleneck link) in an end-to-end path is calculated by sending consecutive pairs of packets. If noncompetitive traffic is present along the path, packets are queued on the slowest (bottleneck) link.*

In the path-flooding method, a measurement is performed by flooding the path links with packets, thus filling up the link capacity. This measurement technique is highly intrusive and heavily affects the communications through the measured links.

timestamped when it is sent. Upon packet arrival at the destination node, the timestamp of consecutive packets is compared with the interarrival time, taking into account the network latency. In order to modulate the rate at which packets are sent, an adaptive algorithm is then used. This measurement method could result in being highly intrusive.

The TCP *simulation* method simulates a TCP connection by means of UDP or ICMP packets. The TCP slow-start phase is simulated as well, in order to determine the link MTU. In this way, the available bandwidth is measured.

In the *path-flooding* method, a measurement is performed by flooding the path links with packets, thus filling up the link capacity. In this way, the real achievable bandwidth at the applicative level is measured, taking into account all the overheads caused by lower-level protocols, the operating system, and the traffic in transit on the links. This measurement technique is highly intrusive and heavily affects the communications through the measured links.

### NETWORK SENSORS

The measurement methodologies described in the previous section can be applied for the implementation of several network sensors [5]. Network sensors are software tools utilized to obtain the desired network performance metric. Network sensors can be classified according to different parameters.

First of all, a sensor could observe the network or probe it; hence, a sensor can perform passive or active measurements. Passive network measurements record the application performance during network utilization. Many efforts have been made to use passive measurements for obtaining network performance information. However, it has been reported that passive network measurement is not always efficient in extracting useful data [8]. Active measurements generate test data, which are sent through the network in order to discover the properties of links and paths. As a drawback, network sensors that use active measurements can generate a great amount of data, thus perturbing the actual value of the parameter under measurement. Intrusiveness states how much the measurement process affects the measure itself. Therefore, a sensor classification can be based on the degree of intrusiveness.

Another classification is based on the degree of cooperation that the sensors require from the network. A first category includes sensors requiring that network locations respond properly to some standard protocol interaction, for instance, ICMP TTL exceeded packets (e.g., traceroute). Another category of sensors needs particular software installed on the locations capable of communicating with the sensors by means of nonstandard protocol, thus requiring a client-server architecture.

### CURRENT MONITORING TOOLS

The Network Weather Service (NWS) [9] is a distributed tool that periodically monitors and dynamically forecasts the performance of various network and computational resources over a given time interval. Currently, the tool includes

sensors for end-to-end TCP/IP performance (achievable bandwidth and latency), available CPU percentage, and available nonpaged memory. Although the NWS sensor interface allows new internal sensors to be configured into the system, the sensors currently included in the NWS are capable of monitoring just the end-to-end TCP/IP performance only. This information could not be sufficient if many sites communicate with each other simultaneously. Indeed, for instance, if two pairs of sites are contemporarily communicating, NWS predicts performance for each pair separately. However, if the communication between the two site pairs shares some common links, the NWS predictions will clearly be too optimistic, as the achievable bandwidth for these links must be shared by the two streams. Thus, information on single link performance must be provided.

TopoMon [10] improves NWS by providing topology, link achievable bandwidth, and link latency information. The network topology and link latency are obtained through *traceroute*. Link achievable bandwidth is obtained from NWS measurements by taking the maximum among the end-to-end achievable bandwidth values of all connections using the considered link.

The study in [11] proposes the utilization of a functional description model for providing a network description. In the model, the network is described from the logical viewpoint without representing all the physical connections between the nodes. The logical network topology consists of a Direct Acyclic Graph (DAG). DAG nodes correspond to either network groups or network hosts. Network group nodes represent the networks connecting a subset of hosts with common network characteristics (e.g., a LAN). The network host nodes correspond to the hosts. The DAG oriented edges represent network group inclusions: network groups can have parent or child network groups and the edges are oriented from a parent network group to a child network group. A child network group represents a subnetwork of its parent network group. The information about how a network node can be reached is obtained by visiting the DAG from its root node. This representation, due to its hierarchical nature, is meant to be more lightweight than a physical network topology description, but the description it provides is incomplete.

The European IST DataGRID project [12] has developed its own architecture for network monitoring. It focuses on collecting the following performance measurements: instantaneous connectivity of an Internet path, packet loss, two-way delay (i.e., RTT), and TCP throughput. The following sensors are used: *PingER* for two-way delay and loss measurements, *UDPmon* for one-way delay and one-way loss measurements, *IPerfER* for TCP throughput (i.e., end-to-end TCP achievable bandwidth), and *Mapcenter* for connectivity checking. Measures are collected using a protocol coordinating different hosts during measurements. The protocol follows a token passing approach. Information about measurements is stored and accessible via the Lightweight Directory Access Protocol (LDAP).

In addition to existing monitoring tools specifically developed for grid computing, several monitoring tools for the Internet exist that either measure network performance or explore network topology [13]. However, their applicability to the grid computing environment is limited, as they only provide data for the nodes involved in the monitoring efforts. In order to have useful information for grid applications, detailed (topology and performance) information about the parts of the global Internet that connect the sites of a grid virtual organization (VO) is needed. Some tools are capable of exploring the topology of LAN installations in great detail. Most prominently, Remos [14] uses SNMP to collect the necessary information. However, SNMP is not available for general Internet connections. The BGP routing protocol provides global topology information, but with the granularity of autonomous systems, which is insufficient for global grid performance optimization purposes.

## CENTRALIZED AND DISTRIBUTED TDS

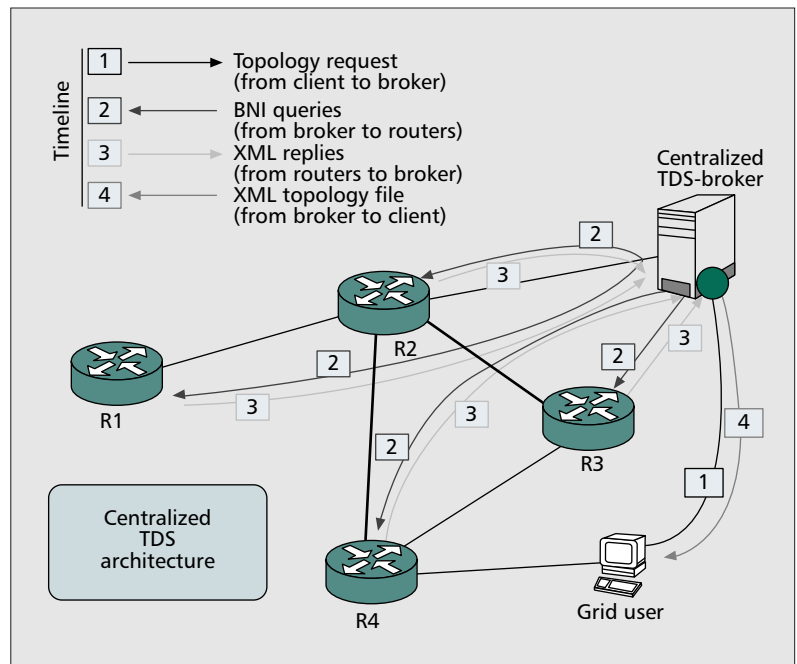
In this section two implementations of a TDS as part of the NIMS are proposed and evaluated: the centralized TDS and the distributed TDS.

### CENTRALIZED TDS

The centralized TDS (C-TDS) is based on the discovery procedure depicted in Fig. 1. Grid users (i.e., service clients) make a topology discovery request to the C-TDS broker through the user-TDS broker interface. Once the user request has been accepted, the C-TDS broker maps the request into a set of queries to be submitted to each router of the grid VO through the TDS broker-router interface, that is, the broker-to-network interface (BNI). Routers' replies are collected by the C-TDS broker, which then elaborates them through an eXtensible Stylesheet Language Transformations (XSLT) engine and builds an eXtensible Markup Language (XML) file containing the requested topology. Eventually the C-TDS broker sends the XML topology file to the users through the user-TDS broker interface. Therefore, in the C-TDS, the users are the information consumers while the C-TDS broker is the information only producer.

The user-TDS broker interface is implemented by means of a bidirectional TCP socket through which client requests are received by the broker, and the XML file, elaborated by the XSLT engine, is sent to the clients. The BNI can be either a standard or a proprietary user-to-network interface (UNI). In the considered C-TDS implementation, the BNI utilizes a proprietary UNI to communicate with the routers. The C-TDS broker submits three queries that to each router concerning the physical interface, the RSVP interface, and the MPLS active label switched paths (LSPs) information, respectively. The XSLT engine performs different XSLT transformations on the XML-based router replies, depending upon the information requested by the users.

The C-TDS provides grid users with both



■ Figure 1. Centralized TDS architecture.

physical and MPLS layer topology information. Both the physical and MPLS layer topologies are represented through an XML file. The physical topology XML file contains the list of routers with their physical interfaces. For each physical interface, the possible logical interfaces, each with its own IP address, MPLS functionalities, RSVP allocated resources, and available bandwidth, are specified. For each interface, the adjacent (bidirectional) interfaces are listed, thus specifying the physical links interconnecting the routers. The MPLS layer topology is an XML file describing the existing LSPs between the routers, their direction (input or output), the router role (ingress, egress, or transit), TE reserved bandwidth, TE load-balance features, explicit routing objects (EROs) (i.e., the list of routers traversed by the LSP), and the LSP source and destination.

### DISTRIBUTED TDS

The topology discovery procedure utilized by the distributed TDS (D-TDS) is depicted in Fig. 2. A user (e.g., a grid host) submits the VO topology discovery request to the D-TDS broker. The broker chooses a set of hosts, within the pool of hosts running the topology discovery service, and contacts them, thus triggering the topology discovery procedure. Each contacted host runs the sensor command. Upon termination of the sensor-based discovery process, each host sends the obtained information to the broker. Upon reception of all the sensor information, the broker merges them and sends an XML reply file containing the discovered topology to the user that requested it. Therefore, in the D-TDS, users are not only information consumers but they participate in producing the requested information.

The D-TDS is based on Globus Toolkit 3.2 (GT3.2) and is deployed in the *Globus-Container* Web server. The communication between the D-

TDS broker and the hosts and among the hosts is implemented via SOAP through the *Globus toolkit* libraries. Both the D-TDS broker and the topology discovery service at the hosts are implemented in Java™. The topology discovery service class has only one remote method, `gettree`, which is called in each host by the broker asynchronously. This asynchronous procedure call has been implemented in Java™ via the *Thread* package. The `gettree` method takes a host list as input (i.e., the set of selected hosts involved in the distributed topology discovery procedure) and runs the *pathchar* sensor toward each host in the list. Upon termination, it returns the tree rooted at the host in which it was called, con-

taining every other host in the list. The tree is obtained by parsing the *pathchar* output through a parser implemented utilizing the *Java StreamTokenizer* package. The hosts return the information to the broker via grid notification. In fact, each host fills up a predetermined service data element (SDE) and sends it to the broker which subscribed to it before calling the `gettree` method. Once the broker has gathered all the information from the selected hosts, it obtains a simple graph from the multigraph (by averaging the values of the duplicated links) and sends it to the hosts that requested the service by means of an XML file.

The D-TDS provides the grid users with the logical topology, that is, the router adjacencies at the IP layer. IP layer adjacencies consist of both physical layer adjacencies (i.e., the adjacencies between physically connected routers) and MPLS layer adjacencies (i.e., two routers non-physically adjacent can be adjacent at the IP layer because they are connected by an LSP).

## RESULTS

The C-DTS and the D-TDS are evaluated in the test-bed depicted in Fig. 3. The test-bed consists of the interconnection of four commercial MPLS routers through two fast Ethernet (FE) interfaces (100 Mb/s), one optical gigabit Ethernet (GbE) interface (1 Gb/s), and one STM1 (ATM over SDH, 155 Mb/s) interface. Three hosts running GT-3.2 are connected to the routers through FE links and represent the clusters connected to the global grid. For assessing the C-TDS capability of obtaining both the physical layer and the MPLS layer topologies in the presence of LSPs, three different LSPs are set up, each one with specific bandwidth reservation.

### TOPOLOGY DELIVERY USE CASES

Figures 4 and 5 show an excerpt of the XML files representing, respectively, the physical layer and the MPLS layer topologies produced by the C-TDS.

The physical topology XML file depicted in Fig.4 shows the description of the fe-0/3/1 interface belonging to the R1 router (address 192.168.1.1) in terms of IP address. Because this physical interface is not adjacent to any other router, adjacency and TE specifications are absent. The subnet address (10.10.21.0/30) specification identifies the attached LAN (represented, in this case, by CLIENT1). The fe-0/3/2 physical interface is adjacent to the fe-1/1/1 belonging to node R2 (192.168.1.2); the fe-0/3/2.0 logical interface (10.10.4.2) is adjacent to fe-1/1/1.0 (10.10.4.1). RSVP tags show input (LSPs "A" and "C") active reservations and reserved bandwidth (50 Mb/s), and output (LSP "B") active reservations and reserved bandwidth (70 Mb/s).

Figure 5 shows an excerpt of the MPLS layer topology obtained by the C-TDS. For each router, LSPs are classified into incoming and outgoing LSPs; in this example the R1 router presents one outgoing and two incoming active LSPs, which are described by node destination, name, EROs, bandwidth reserved for the LSP

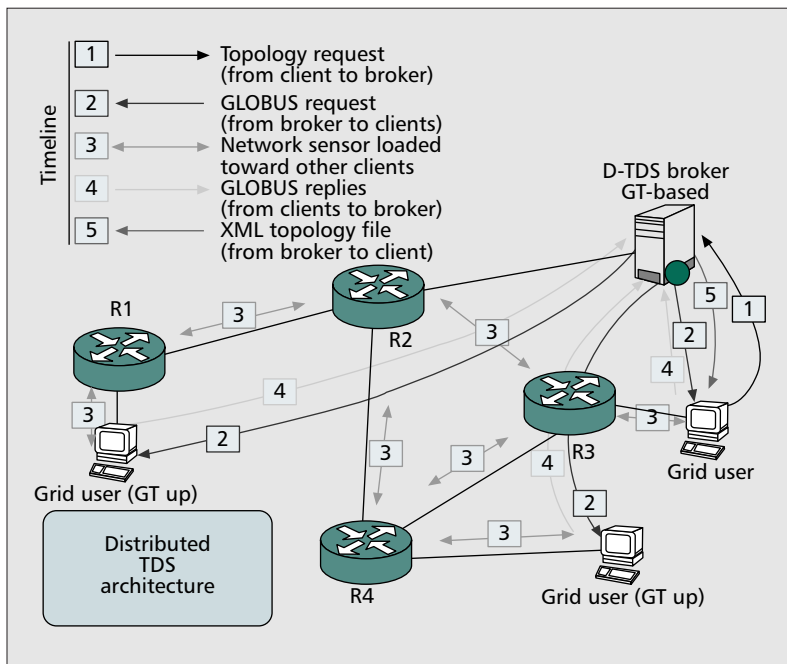


Figure 2. Distributed TDS architecture.

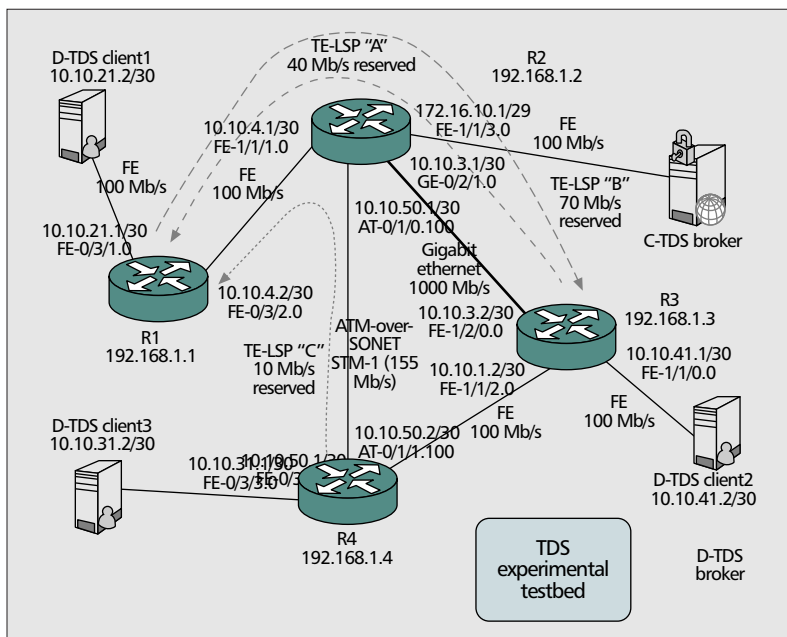


Figure 3. The testbed.

(i.e., TE bandwidth) and load-balancing option (this option is valid just for the outgoing LSP).

The dimension of the files exchanged between C-TDS broker and routers is in the order of tens of kB. In particular, the biggest among the XML-based router replies in terms of file size is the one related to the interface information query. Its dimension depends on the number of physical interface cards (PICs) installed on each router. For example, in the considered testbed, the most connected node R2, which features four physical interfaces and eight internal PICs, generates a 16 kB XML interface file.

An excerpt of the D-TDS logical network topology is depicted in Fig. 6. To obtain the formatted XML file by merging the received host trees, the broker is provided with information regarding which interfaces (i.e., IP addresses) belong to which routers. In addition, the broker assumes fully internal interconnection between the router interfaces. In the XML file some tag “nodes” which represent the IP of the selected hosts are present. Following the node information, the list of all the links appears; it includes the measured link bandwidth, RTT, queue delay, and dropped packets, computed by *pathchar*. As can be seen in Fig. 6, the D-TDS is able to obtain an estimate of the link bandwidth and of the RTT (on which the latency depends). However, the D-TDS is not able to obtain information on each router interface that is as specific as the information obtained by the C-TDS. In addition, the link bandwidth appears underestimated, especially for high-capacity links, due to the *pathchar* specific implementation.

### TOPOLOGY DELIVERY PERFORMANCE

The *topology delivery time* is the main parameter utilized to evaluate the performance of both the C-TDS and the D-TDS. The topology delivery time is defined as the time elapsing from the topology request until the topology information delivery.

In the C-TDS the topology delivery time depends on the time necessary to collect the XML-based router replies from each router and to perform the XSLT file processing. In the considered experimental setup, the collection of the XML-based router replies requires about 1 s and slightly depends on the level of details of the required information. In particular, for each router, 800 ms are necessary for the C-TDS broker to login into the router and 200 ms are necessary for the exchange of the three request-reply messages. The XSLT engine processing time is 100 ms. Therefore, if the C-TDS broker queries the router in series, the topology delivery time is about 4.1 s. However, the parallelization of the router queries reduces the topology delivery time to 1.1 s.

In the D-TDS the topology delivery time depends on the number of hosts involved in the discovery procedure, the network size, and the level of detail of the required information. In general, considering a network with diameter  $D$ , a host set containing  $H$  hosts, and a fixed probe running time per hop, the time necessary to build the tree information at each host is proportional to  $O((H - 1)D)$ . Indeed, measurements are parallelized on the basis of

```
<physical-topology>
  <node >
    <id-address>192.168.1.1</id-address>
    <physical-interface>
      <name>fe-0/3/1</name>
      <bandwidth>100mbps</bandwidth>
      <logical-interface>
        <name>fe-0/3/1.0</name>
        <type>mpls</type>
        <IPAddress>10.10.21.1</IPAddress>
        <subnet-address>10.10.21.0/30</subnet-address>
        .....
      </physical-interface>
    <physical-interface >
      <name>fe-0/3/2</name>
      <bandwidth>100mbps</bandwidth>
      <destination-node>192.168.1.2</destination-node>
      <dest-physical-interface>fe-1/1/1</dest-physical-interface>
      <logical-interface>
        <name>fe-0/3/2.0</name>
        <type>mpls</type>
        <IPAddress>10.10.4.2</IPAddress>
        <subnet-address>10.10.4.0/30</subnet-address>
        <dest-logical-interface>fe-1/1/1.0</dest-logical-interface>
        <IP-destination>10.10.4.1</IP-destination>
        <RSVP-TE-link>
          <IN-active-resv>2</IN-active-resv>
          <IN-total-resv-bw>50Mbps</IN-total-resv-bw>
          <IN-available-bw>50Mbps</IN-available-bw>
          <OUT-active-resv>1</OUT-active-resv>
          <OUT-total-resv-bw>70Mbps</OUT-total-resv-bw>
          <OUT-available-bw>30Mbps</OUT-available-bw>
        </RSVP-TE-link>
      </logical-interface>
    </physical-interface>
    .....
  </node>
```

■ Figure 4. XML physical topology excerpt.

the hosts running the measurement and performed on a per-hop basis. However, no further parallelization is exploitable (i.e., running different instances of *pathchar* toward different hosts in parallel on the same host) because different measurements could interfere with each other. The topology delivery time is obtained by summing the time necessary to build the host trees and the time the broker needs for merging the received host trees into a single topology graph. In the considered experimental setup the topology delivery time is about 4 min. During the measurements, the network is not subject to congestions due to *pathchar* low invasiveness.

### DISCUSSION

The proposed C-TDS and D-TDS present advantages and drawbacks. Since the C-TDS directly queries the routers, one advantage of the C-TDS is the capability of obtaining the complete physical layer and MPLS layer topologies, regardless of the particular distributed routing protocol utilized. In addition, the level of details of the information related to the routers can be tuned by suitably selecting the router queries. Finally, the utilization of XML and XSLT provides modularity, allowing future extensions by implementing new BNI queries and new XSLT files. One of the main drawbacks of the centralized approach is the need for the C-TDS broker to

```

<MPLS-topology>
  <node>
    <id-address>192.168.1.1</id-address>
    <output-link>
      <lsp-name>B</lsp-name>
      <to-node>192.168.1.3</to-node>
      <to-interface>10.10.3.2</to-interface>
      <from>192.168.1.1</from>
      <TE-load-balance>random</TE-load-balance>
      <TE-bw>70Mbps</TE-bw>
      <ERO-node>10.10.4.1</ERO-node>
      <ERO-node>10.10.3.2</ERO-node>
    </output-link>
    <input-link>
      <lsp-name>A</lsp-name>
      <to>10.10.4.2</to>
      <TE-bw>40Mbps</TE-bw>
      <intermediate-node>10.10.3.2</intermediate-node>
      <intermediate-node>10.10.4.1</intermediate-node>
    </input-link>
    <input-link>
      <lsp-name>C</lsp-name>
      <to>10.10.4.2</to>
      <TE-bw>10Mbps</TE-bw>
      <intermediate-node>10.10.50.1</intermediate-node>
      <intermediate-node>10.10.4.1</intermediate-node>
    </input-link>
  </node>
  .....

```

■ **Figure 5.** XML MPLS layer topology excerpt.

have administrative privileges on the routers. Thus, the TDS must be implemented by the network providers. In addition, if the required information involves more than one administrative domain, a multi-tier architecture based on a hierarchy of C-TDS brokers must be implemented to avoid interdomain issues. Furthermore, the C-TDS provides the information on the bandwidth reserved on the router interfaces by the LSPs (i.e., the TE bandwidth) based on the RSVP reservation made at the routers but it does not provide real-time information on the link bandwidth utilization. Hence, the C-TDS can be considered a resource discovery service rather than a real-time monitoring tool. Moreover, the C-TDS broker represents a critical and potential central point of failure requiring the implementation of backup brokers.

The D-TDS presents some important differences with the C-TDS. An advantage of the D-TDS is that the discovered topology includes also the links between the hosts and the routers; thus, the access network is included in the estimation procedure. In addition, the D-TDS does not incur interdomain issues because it exploits IP layer probes. An important aspect for the D-TDS is the algorithm utilized for merging the single-host topology viewpoints (i.e., the host trees) into a unified picture (i.e., the logical topology). The algorithm must deal with the issues of links appearing in more than one host view (duplicated links between the same router interfaces) and of resolving which interfaces belong to the same router. The former problem can be solved, as in the proposed implementation, by collapsing the duplicated links into a single link. Quantitative information about the link

such as bandwidth and latency can then be obtained, for example, by averaging the set of values reported. An automatic way of solving the latter issue is to incorporate the D-TDS the approach proposed in [15] in order to overcome the so-called problem of router aliases. A drawback of the D-TDS architecture is that it does not always guarantee the discovery of the complete VO network. Indeed, the host set or the host placement could not suffice for guaranteeing the complete network topology discovery. For example, in Fig. 3, if the hosts in the set are connected to routers R1, R2, and R3, the link R4-R2 will not be discovered by the D-TDS. Instead, if the hosts are connected to the routers R1, R3, and R4, all the network links will be discovered. On the other hand, the distributed approach adopted in the D-TDS also represents an advantage. Indeed, the possibility of selecting the number of hosts involved in the topology discovery process allows us to trade the level of details of the discovered topology with the amount of resources (i.e., messages) utilized for the discovery. This makes the D-TDS scalable to a VO with a large number of network elements.

## SUMMARY

This article has reviewed current performance metrics, measurement methodologies, sensors, and tools to be considered for implementing the global grid network monitoring and information service (NIMS). In addition, two different approaches for implementing a grid topology discovery service (TDS) have been presented: the centralized TDS (C-TDS) and the distributed TDS (D-TDS). The TDS provides either a grid computing application or the middleware with the capability of retrieving information on the grid network infrastructure status. On the one hand, the C-TDS is able to obtain in a few seconds very detailed information about the status of each single network interface, but router administrative privileges are required in order to retrieve this information. On the other hand, the D-TDS allows us to obtain information on the topology traversed by IP packets, that is, the logical topology. The D-TDS is scalable and it utilizes tools available to generic grid users, such as pathchar. However, the D-TDS requires minutes to provide the requested information and the level of detail of the discovered topology information depends on the number of grid hosts involved in the discovery procedure and on their placement.

## ACKNOWLEDGMENTS

The author would like to thank Riccardo Vestrini for his helpful suggestions and support on XSLT.

This work has been supported by the Italian Ministry of University and Research (MIUR) under the FIRB "GRID.IT" Project.

## REFERENCES

- [1] G. Clapp *et al.*, "Grid Network Services," Informational, Grid High-Performance Networking Research Group (GHPN-RG), May 2005, available at [forge.grwdforum.org/projects/ghpn-rg](http://forge.grwdforum.org/projects/ghpn-rg) as draft-ggf-ghpen-netserve-2.0.
- [2] T. Ferrari (Ed.), "Grid Network Services Use Cases," Informational, Grid High-Performance Networking Research Group (GHPN-RG), Aug. 17, 2005.

- [3] F. Baroncelli et al., "A Service Oriented Network Architecture suitable for Global Grid Computing," *Proc. ONDM 2005*, Milan, Italy, Feb. 7–9, 2004.
- [4] M. Danelutto, "Adaptive Task Farm Implementation Strategies," *Proc. 12th Euromicro Conf. Parallel, Distributed and Network-Based Processing*, 2004, Feb. 11–13, 2004, pp. 416–23.
- [5] F. Michaut and F. Lepage, "Application-Oriented Network Metrology: Metrics and Active Measurement Tools," *IEEE Commun. Surveys*, vol. 7, no. 2, 2nd Quarter, 2005.
- [6] R. Prasad et al., "Bandwidth Estimation: Metrics, Measurement Techniques, and Tools," *IEEE Network*, vol. 17, no. 6, Nov.–Dec. 2003, pp. 27–35.
- [7] K. Lai and M. Baker, "Measuring Link Bandwidths Using A Deterministic Model Of Packet Delay," *Proc. ACM SIGCOMM Conf. 2000*, 283–94.
- [8] K. C. Klay and S. McCreary, "Internet Measurement And Analysis: Passive and Active Measurements," 1999, available at <http://www.caida.org/outreach/papers/1999/Nae4hansen>
- [9] R. Wolski, "Experiences with Predicting Resource Performance On-line in Computational Grid Settings," *ACM SIGMETRICS Perf. Evaluation Rev.*, vol. 30, no. 4, Mar. 2003, pp. 41–49.
- [10] M. D. Burger, T. Kielmann, and H. E. Bal, "TopoMon: A Monitoring Tool for Grid Network Topology," *Proc. Int'l. Conf. Computational Science (ICCS 2002)*, Amsterdam, Apr. 21–24, 2002
- [11] S. Lacour, C. Pérez, and T. Priol, "A Network Topology Description Model for Grid Application Deployment," *Proc. 5th IEEE/ACM Int'l. Wksp. Grid Computing (GRID 2004)*, Pittsburgh, PA, USA, pp. 61–68, Nov. 2004.
- [12] P. Vicat-Blanc Primet, R. Harakaly, and F. Bonassieux, "Grid Network Monitoring in the European DataGrid Project," *Int'l. J. High Performance Computing Applications*, vol. 18, no. 3, Fall 2004, pp. 293–304.
- [13] P. Francis et al., "IDMaps: A Global Internet Host Distance Estimation Service," *IEEE/ACM Trans. Net.*, v. 9, no. 5, Oct. 2001.
- [14] P. Dinda et al., "The Architecture of the Remos System," *Proc. IEEE Symp. High Performance Distributed Computing (HPDC10)*, San Francisco, CA, 2001.
- [15] R. Govindam and H. Tangmunarunkit, "Heuristics for Internet map discovery," *Proc. INFOCOM 2000*, vol. 3, 26–30 Mar. 2000, pp. 1371–80.

## BIOGRAPHIES

LUCA VALCARENGHI (valcarenghi@sssup.it) holds a Laurea degree in electronics engineering (1997) from Politecnico di Torino, Italy, an M.S. in electrical engineering (1999) and a Ph.D. in electrical engineering/telecommunications (2001), both from the University of Texas at Dallas. Since September 2002 he has been an assistant professor at the Scuola Superiore Sant'Anna of University Studies and Doctoral Research, Pisa, Italy. He co-authored more than 45 papers published in international journals and presented in leading international conferences. His main research interests are optical networks design, analysis, and optimization; artificial intelligence optimization techniques; communication network reliability; IP over WDM networking; and QoS in network infrastructures for grid computing.

LUCA FOSCHINI (foschini@sssup.it) holds a B.S. in computer science engineering from the University of Pisa and the Sant'Anna School for Advanced Studies. He is currently pursuing an M.S. in computer science engineering at the aforementioned institutions. He is now a visiting student at the University of California at Santa Barbara, researching intrusion detection systems and working as a research intern for Ask Jeeves, where he does research on document clustering.

FRANCESCO PAOLUCCI (fr.paolucci@sssup.it) holds a Laurea degree in telecommunications engineering (2002) from the University of Pisa. He is now a Ph.D. student in telecommunications engineering at the Scuola Superiore Sant'Anna of University Studies and Doctoral Research in Pisa, Italy. His

```
<D-TDS-topology>
  <node>
    <name>10.10.21.2</name>
    <dns-name>CLIENT1</dns-name>
    <node-type>host</node-type>
    <link>
      <source>10.10.21.2</source>
      <destination>10.10.21.1</destination>
      <dns-destination>R1</dns-destination>
      <timestamp>12:34</timestamp>
      <available-bw>50 Mb/s</available-bw>
      <rtt>7.1</rtt>
      <dropped>5</dropped>
      <queue-delay>1.3E-4</queue-delay>
    </link>
  </node>
  <node>
    <name>217.9.70.112</name>
    <dns-name>R1</dns-name>
    <node-type>router</node-type>
    <link>
      <source>10.10.21.1</source>
      <destination>10.10.21.2</destination>
      <dns-destination>CLIENT1</dns-destination>
      <timestamp>12:34</timestamp>
      <available-bw>50 Mb/s </available-bw>
      <rtt>7.1</rtt>
      <dropped>5</dropped>
      <queue-delay>1.3E-4</queue-delay>
    </link>
    <link>
      <source>10.10.4.2</source>
      <destination>10.10.4.1</destination>
      <dns-destination>R2</dns-destination>
      <timestamp>12:33</timestamp>
      <available-bw>87 Mb/s</available-bw>
      <rtt>7.43</rtt>
      <dropped>0</dropped>
      <queue-delay>1.6E-4</queue-delay>
    </link>
  </node>
  .....
```

■ Figure 6. Excerpt of the D-TDS logical topology output file.

main interests are communication network reliability, QoS in network infrastructure for grid computing, MPLS/GMPLS architectures, and traffic engineering protocol extensions, interdomain routing, and WDM network design.

FILIPPO CUGINI (filippo.cugini@cnit.it) received a Laurea degree in telecommunication engineering from the University of Parma, Italy. Since 2001 he is a research engineer at the CNIT National Laboratory of Photonic Networks, Pisa, Italy. His main research interests include MPLS and GMPLS protocols and architectures, survivability in IP over WDM networks, and traffic engineering in grid networking.

PIERO CASTOLDI (castoldi@sssup.it) is an associate professor in telecommunications at the Center of Excellence for Communication Networks Engineering of Scuola Superiore Sant'Anna. He is also director of the CNIT National Laboratory of Photonic Networks. His scientific activity has mainly covered the area of digital transmission, wired and wireless networks, and, more recently, design, resilience schemes, and service platforms for metro and core networks. He has authored more than 70 technical papers and an international book on CDMA.