

# Implementation of a GMPLS-based Network with End Host Initiated Signaling

X. Zhu, X. Zheng, M. Veeraraghavan  
University of Virginia

Z. Li, Q. Song, I. Habib  
City University of New York

N. S. V. Rao  
Oak Ridge National Laboratory

*Abstract<sup>1</sup> -- In this article, we describe our experiences in implementing an experimental wide-area GMPLS network called CHEETAH (Circuit-Switched End-to-End Transport Architecture). The key concept is to add a complementary end-to-end circuit based service with dynamic call-by-call bandwidth sharing to the connectionless service already available to end hosts via the Internet. The current CHEETAH experimental network consists of off-the-shelf GMPLS-capable SONET switches (with Ethernet interfaces) deployed at three locations, Research Triangle Park, North Carolina, Atlanta, Georgia, and Oak Ridge, Tennessee. We designed and implemented a CHEETAH software package to run on Linux end hosts connected to the CHEETAH network. Among other functions, this software package includes an RSVP-TE client module to enable end users and applications to dynamically initiate requests for dedicated end-to-end circuits and receive/respond to requests for circuits. We present measurements for typical end-to-end circuit setup delays across this network. For example, end-to-end circuit setup delay from a Linux end host in NC to a host in Atlanta is 166ms.*

## I. INTRODUCTION

Large-scale eScience projects from fields as diverse as earth science, high energy and nuclear physics, astrophysics, molecular dynamics, and genomics typically involve teams of scientists from different fields who are geographically distributed across the country. Networking capabilities that enable the transfers of massive data sets, remote visualizations and control of computations and experiments, can significantly enhance the productivity of these teams. Current networks do not provide these capabilities because of limited bandwidth and a lack of support to control delay/jitter.

Recent advances in optical networking technologies hold an unprecedented potential to provide bandwidth on-demand services. Leveraging these advances, we proposed a networking concept called CHEETAH to address the needs of a broad class of applications included among which are eScience applications [1]. CHEETAH, which stands for “Circuit-switched High-speed End-to-End Transport Architecture,” is a networking solution in which an optical circuit-switched network is built as an “**add-on**” to the connectionless Internet. The goal of CHEETAH is to provide end-host applications access to end-to-end connection-oriented services, while preserving the connectionless services already available to them.

To provide such end-to-end connection-oriented services in a cost-effective manner, dynamic sharing of bandwidth is critical. A key set of control-plane protocols have been defined

under the umbrella term “Generalized Multi-Protocol Label Switching (GMPLS)” to enable such dynamic sharing of bandwidth on a call-by-call basis in connection-oriented networks [2]. One component of GMPLS control-plane protocols is signaling. Signaling protocols are required for a distributed implementation of dynamic bandwidth sharing. A commonly implemented GMPLS signaling protocol is Resource Reservation Protocol - Traffic Engineering (RSVP-TE) [3].

Among the different types of connection-oriented networks supported by GMPLS protocols are Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) networks, Wavelength Division Multiplexed (WDM) networks, and Space Division Multiplexed (SDM) networks. Given our goal to support a broad class of applications and consequently a high granularity in per-circuit capacities, we chose SONET switches for the CHEETAH network.

Our choice of SONET was further driven by the availability of network equipment capable of mapping Ethernet frames into SONET frames, which enabled us to achieve our goal of providing end-host applications with end-to-end circuits. These circuits would thus consist of Ethernet segments at the edges mapped to SONET circuits (carrying the Ethernet signals transparently) through metro-/wide-area segments. Combined with GMPLS control-plane protocols, we had the main ingredients necessary to achieve our goals for CHEETAH.

In addition to these ingredients, we identified the need for new software to run on CHEETAH end hosts, which we refer to as “CHEETAH end-host software.” For example, we require an RSVP-TE software module at the end hosts to initiate the setup of a circuit upon demand from applications. Wherever possible, our goal is to integrate libraries of this CHEETAH end-host software into application software modules to make CHEETAH service transparent to users (humans).

This paper describes an implementation of our CHEETAH concept. We designed and deployed a wide-area CHEETAH experimental network across Research Triangle Park, NC, Atlanta, GA, and Oak Ridge, TN. We have implemented a first version of the CHEETAH end-host software package for Linux PCs. In this paper, we provide details about the RSVP-TE client module of this package, and describe how it is used to dynamically initiate the setup and teardown of end-to-end circuits across the CHEETAH network. We also provide measurements of the end-to-end call setup delay, which is a critical overhead component in connection-oriented networks.

1. This work was carried out under the sponsorship of NSF ITR-0312376, NSF ANI-0335190, NSF ANI-0087487, and DOE DE-FG02-04ER25640 grants.

The rest of the paper is organized as follows. Section II reviews GMPLS protocols, summarizes the CHEETAH concept and related work. Section III describes the wide-area CHEETAH experimental network. We describe the CHEETAH end-host software package in Section IV. Section V describes the signaling performance experiments we conducted on the CHEETAH network. We conclude the paper in Section VI.

## II. BACKGROUND AND RELATED WORK

### A. GMPLS control-plane protocols

There are three major components in GMPLS control-plane protocols: RSVP-TE signaling protocol, Open Shortest Path First - Traffic Engineering (OSPF-TE) routing protocol [4], and Link Management Protocol (LMP) [5]. These three protocols are designed to be implemented in a control processor associated with each network switch to provide corresponding functionality for its switch. Each of these protocols provides an increasing degree of automation, and correspondingly decreasing dependence upon manual network administration.

The main purpose of the LMP module is to automatically establish and manage the control channels between adjacent nodes, to discover and verify data-plane connectivity, and to correlate data-plane link properties. The purpose of the OSPF-TE routing protocol software module located at a switch is to enable the switch to send topology, reachability and loading conditions of its interfaces to other switches, and receive corresponding information from other switches. The purpose of the RSVP-TE signaling engine at a switch is to manage the bandwidth of all the interfaces on the switch, and to program the data-plane switch hardware to enable it to forward demultiplexed incoming user bits or packets as and when they arrive. A typical RSVP-TE engine implemented in switches executes five steps when it receives a connection setup request (i.e., an RSVP-TE PATH message): signaling message parsing, route lookup, connection admission control, data-plane configuration (i.e., switch fabric configuration), and signaling message construction.

### B. CHEETAH Concept

The CHEETAH architecture is illustrated in Fig. 1. End hosts are equipped with two Ethernet Network Interface Cards (NICs). The primary NICs (NIC1) in the end hosts are connected to the public Internet through the usual LAN Ethernet switches and/or IP routers, while the secondary NICs (NIC2) are connected to Ethernet ports on Ethernet-to-SONET circuit gateways. Ethernet-to-SONET circuit gateways, in turn, are connected to wide-area SONET circuit-switched networks, in which both circuit gateways and pure SONET switches are equipped with GMPLS protocols to support call-by-call dynamic bandwidth sharing. Dynamic bandwidth sharing through a distributed mechanism, such as the one offered by GMPLS control-plane protocols, is critical to achieving scalability, an important factor for any network as noted in [6].

End-to-end CHEETAH circuits (as shown with dashed

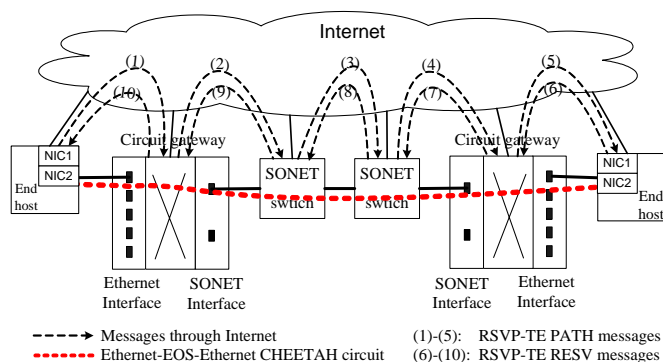


Figure 1. CHEETAH concept

lines in Fig. 1) are set up dynamically between end hosts with RSVP-TE signaling messages being processed at each intermediate gateway/switch in a hop-by-hop manner.

CHEETAH is designed as an “add-on” service to existing Internet connectivity, leveraging the services of the latter. This design brings the following benefits: (i) The connection to the Internet allows a CHEETAH end host to communicate with other non-CHEETAH hosts on the Internet while it is in a communication session with another CHEETAH end host through a dedicated CHEETAH circuit; (ii) Signaling messages can be transmitted through the primary NICs and the Internet, without requiring a dedicated signaling channel; (iii) Applications can selectively choose to request CHEETAH circuits only when the Internet path is estimated to provide a lower service quality than the CHEETAH circuit, and further “fall back” to the Internet path if the CHEETAH circuit setup attempt fails due to an unavailability of circuit resources on the CHEETAH network. The latter allows for a gradual growth of the CHEETAH network.

### C. Related work

Other optical connection-oriented testbeds being deployed in support of eScience applications include CANARIE’s CA\*net 4 [7], OMNInet [8], SURFnet [9], UKLight [10], DOE’s UltraScience net [11], and NSF DRAGON network [12]. Several research efforts within these projects are focused on how to provision circuits. For example, the User Controlled LightPaths (UCLP) project [13], supported by CANARIE, developed software to allow end users to provision inter-switch/router SONET circuits and to create their own desired high-level network topologies (e.g. IP network topologies), which are held for long durations. The NSF-funded DRAGON project is based on GMPLS control-plane protocols and enables the dynamic creation of (long-held) application-specified network topologies to support large-scale eScience applications. The CHEETAH project goals differ from these projects primarily in its goals to enable large-scale connection-oriented networks, which drives it toward supporting short-duration calls triggered automatically by application software such as FTP and web clients/servers in addition to its support of eScience projects.

### III. CHEETAH NETWORK HARDWARE AND CONNECTIVITY

This section describes the wide-area CHEETAH experimental network.

#### A. Network switches and hosts

We selected the Sycamore SN16000 Intelligent Optical Switch for the primary network nodes in the CHEETAH network. Our reasons for doing so were **two-fold**. **First**, the SN16000 switch supports both SONET and Ethernet interface cards. In the current implementation, the SN16000 switch simply treats Ethernet ports as space-division multiplexed ports (while the SONET ports are time-division multiplexed). Therefore the Ethernet-to-SONET mapping action can be viewed as a circuit-switching action in which an Ethernet port is cross-connected to a specified number of time slots on a SONET interface. The Ethernet-to-SONET mapping mechanism is a standardized procedure called Generic Framing Procedure (GFP) [14]. In addition, Virtual Concatenation, also a standardized mechanism, is used to select the SONET rate to be as closely matched to the Ethernet rate as possible [15]. For example, a 1Gb/s Ethernet signal can be mapped to a 21-OC-1 virtually concatenated SONET signal instead of an OC-48 (2.5Gb/s) signal, which would have been the appropriate level to use in the original SONET hierarchy.

**Second**, the Sycamore SN16000 switch has a robust implementation of GMPLS control-plane protocols, RSVP-TE and OSPF-TE. It supports the major GMPLS control-plane-related RFCs: RFC 3473, 3477, 3945, 3946 for RSVP-TE, and RFC 3630 for OSPF-TE. While these specifications specify procedures for homogeneous circuits (e.g., pure-SONET, where the incoming and outgoing line cards used on the circuit are both SONET), they do not support hybrid circuits, i.e., Ethernet-SONET circuits. To establish such circuits, most equipment require administrators to use network management software. Standardization is underway to extend RSVP-TE to handle such hybrid circuits. Meanwhile, Sycamore provided CHEETAH a version of their control-plane software, which implements a proprietary solution for RSVP-TE control of such hybrid circuits. The built-in RSVP-TE software at the SN16000 switches makes these switches easy to use. We simply configure parameters at each switch to specify the data and control-plane addresses of the remote ends. Our end-host RSVP-TE software inter-operates with the built-in RSVP-TE engines of these switches.

The **client hosts** on the CHEETAH network are general-purpose Linux PCs equipped with two Ethernet NICs, the primary NIC (which can be copper or optical) for connectivity to the Internet, and the secondary NIC, an optical 1GbE or 10GbE card, to connect to an SN16000 switch.

#### B. Connectivity

##### B.1. Data-plane

Fig. 2 shows the data-plane topology of the wide-area CHEETAH experimental network. Three CHEETAH Point of

Presences (PoPs) equipped with SN16000 switches are located at MCNC in Research Triangle Park (RTP), NC, Southern Crossroads (SOX)/Southern Light Rail (SLR) in Atlanta, GA, and Oak Ridge National Laboratory (ORNL) in Oak Ridge, TN.

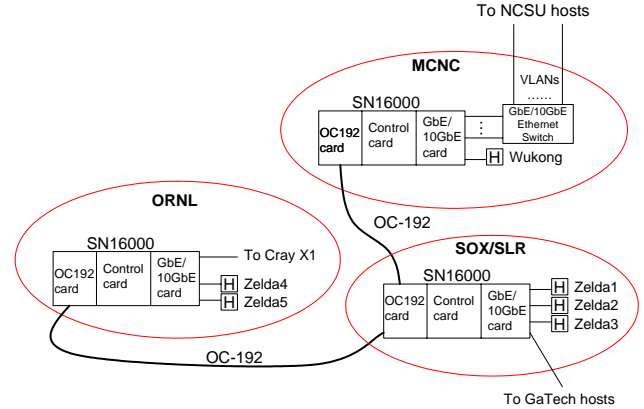


Figure 2. Wide-area CHEETAH testbed topology (data-plane)

The three SN16000 switches are interconnected by long-distance OC-192 SONET circuits. Each of the three SN16000 switches is equipped with GbE interface cards (each with ten ports), which are used to connect CHEETAH client hosts' secondary NICs (data-plane interfaces). Fig. 2 shows that a few client hosts, *Wukong* in MCNC, *Zelda1*, *Zelda2*, *Zelda3* in SOX/SLR, and *Zelda4* and *Zelda5* in ORNL, which are physically located at the three CHEETAH PoPs and hence directly connected to the SN16000 switches (set up for experimental research). Since it is not feasible to connect hosts resident in enterprises/campuses that are geographically distant from the CHEETAH PoPs, we use Virtual Local Area Networks (VLANs) or MultiProtocol Label Switch (MPLS) connections. For example, in Fig. 2, multiple GbE VLANs are set up through GbE Ethernet switches at NCSU and MCNC. Each VLAN connects one end host at NCSU to one GbE port on the SN16000 switch in MCNC.

##### B.2. Control-plane

The **SN16000** switch supports two types of interfaces for transmitting control-plane messages: (i) in-band Data Communication Channels (DCC) embedded within SONET data-plane interfaces, and (ii) out-of-band Ethernet control ports. We choose the out-of-band signaling option in the CHEETAH network for switch-to-switch control-plane communication simply because we needed the ability to capture control-plane messages for experimental reasons. As shown in Fig. 3, we connected the Ethernet control ports of the switches to the Internet. Control-plane communication between end hosts and switches are also carried on the Internet, with the primary NICs on the end hosts being used for this connectivity.

On the face of it, setting up the control-plane connectivity seemed to be as simple as merely connecting end hosts and control ports of switches into the Internet and using the latter for control-plane messages. However, the actual solution

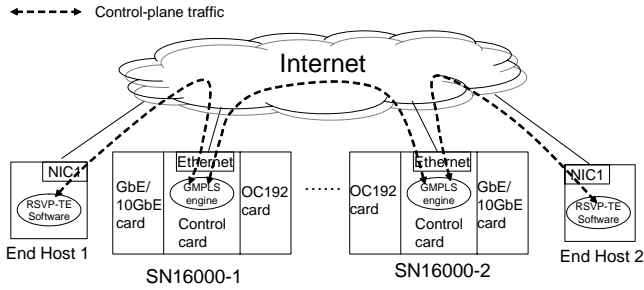


Figure 3. CHEETAH control-plane architecture

turned out to be much more complex. For example, many enterprises use private IP addresses for end hosts, (e.g., ORNL on the CHEETAH network) or dynamically assign IP addresses to end hosts. Private/dynamic IP addresses pose no problem if all applications run in “client” mode; however, additional mechanisms are required for address discovery if end hosts run applications in a “server” mode. This is because a server needs to be reachable. The end-host RSVP-TE software on CHEETAH end hosts is essentially a “server” module for incoming calls because it needs to be reached. We address this issue by building and using IP-in-IP tunnels between SN16000s and end hosts.

Next consider the SN16000’s connectivity to the Internet. For reliability reasons, the SN16000 control card has two Ethernet control ports. Being an experimental network, we connect only one port. IP-in-IP tunnels are needed between any pair of adjacent SN16000 switches to establish OSPF adjacency. These tunnels are also used for RSVP-TE signaling.

### C. Security

Securing control-plane access to the Sycamore switches is important to protect them from Internet attacks, such as Denial-of-Service (DoS) attacks, prevent hackers from accessing the control processors of these switches, and prevent malicious users from sending RSVP-TE PATH messages to tie up large amounts of bandwidth.

To provide protection from such attacks, the CHEETAH network implements a combined security mechanism consisting of firewall, DoS protection, and IPsec Virtual Private Network (VPN) services. To provide these services, we selected the Juniper Netscreen-5XT (NS-5XT) [16] device. One such device is placed in front of the Ethernet control port of each SN16000 switch. Fig. 4 illustrates this CHEETAH network security solution.

Firewall settings are configured in the NS-5XT devices to prevent unwanted traffic reaching the Ethernet control ports of the SN16000s. IPsec tunnels are configured at the NS-5XTs to secure switch-to-switch control-plane messages, such as RSVP-TE and OSPF-TE messages. To secure host-to-switch control-plane messages, we installed an open-source IPsec Linux software package called Openswan [17] on all CHEETAH end hosts. IPsec tunnels are created between hosts and NS-5XTs to protect the RSVP-TE messages.

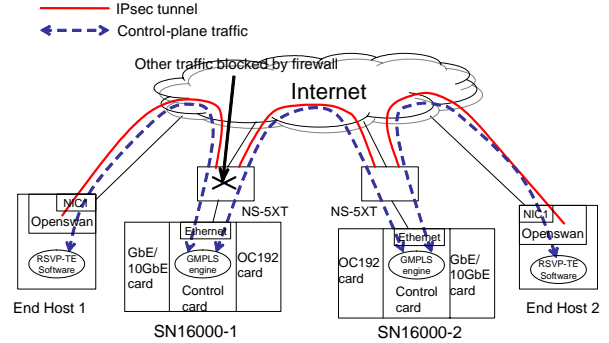


Figure 4. CHEETAH security architecture

Security functions provided with this solution are host-level authentication, integrity and privacy. All RSVP-TE messages are encrypted and encapsulated with an IPsec header. A CHEETAH network administrator can be assured that only users of end hosts to which IPsec tunnels are created from the SN16000 switches can initiate applications that automatically request and reserve bandwidth. User-level authentication and authorization is not provided in the current CHEETAH security solution but will be considered in the next release. In our opinion, fine-grained authorization and accounting are better handled with enhancements to the switch control-plane software to trigger database lookups upon receiving PATH messages (e.g., to verify if a given end host has the authority to request a certain level of bandwidth) as is currently done in the telephone network. Our plan is to work with the switch vendor to include such a functionality in a future GMPLS code release.

Data-plane security is left up to the applications that use CHEETAH circuits. For example, if secure data transfers are required, users can run SFTP or GridFTP [18].

## IV. CHEETAH END-HOST SOFTWARE

The CHEETAH end-host software architecture is shown in Fig. 5. The **Optical Connectivity Service (OCS) client** module answers queries as to whether or not the correspondent end host (called party) has access to the CHEETAH network. It does this sending a TXT query to a Domain Name Server (DNS). The TXT resource record is a generic type supported by DNS to allow users to store any data about hosts. The TXT data we store for CHEETAH end hosts consists of an indication that it is a CHEETAH end host, along with the IP and MAC addresses of the host’s secondary (data-plane) NIC. Recall the choice of two paths available to CHEETAH end hosts as described in Section II.B. The **routing decision** module answers queries from applications on whether or not to attempt a circuit setup. The **RSVP-TE** module is used to request circuit setup and teardown. The **Circuit-TCP (C-TCP)** [19] module is a transport protocol for CHEETAH circuits.

In this section, we focus on the RSVP-TE module. Readers may refer to [20] for more details about other modules.

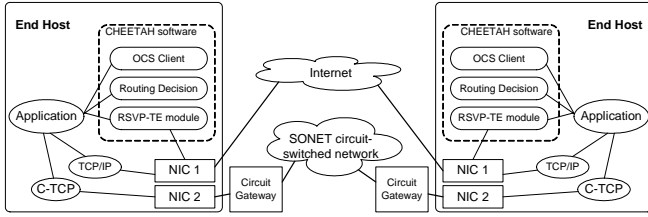


Figure 5. CHEETAH end-host software

#### A. Functional description of the RSVP-TE module

Recall from Section II.A that switches perform the following five steps upon receiving a PATH message: message parsing, route lookup, connection admission control, data-plane configuration, and message construction. Similar functions are needed in the **end-host RSVP-TE module**. The **message parsing and construction** steps are the same on end hosts as in the switches. Let us focus on the remaining three steps.

We treat **route lookup** at an end host in the same manner as is currently in practice with IP packet forwarding, in that end hosts are simply programmed with a default gateway address and all IP packets, destined to nodes outside the local network, are sent to this gateway. The RSVP-TE module at the end host sends all PATH messages, without providing an Explicit Route Object (ERO), to its circuit gateway. The circuit gateway then either performs source route computation to reach the specified destination and adds an ERO to the PATH message, or simply determines the next-hop switch toward which to direct the call setup. This mode of operation, in which the end client does not include an ERO, is called the GMPLS overlay model [21] in contrast to the peer model where the end client is a router that computes the ERO itself.

**Connection admission control (CAC)** on end hosts is relatively simple. Given the SN16000 does not currently support VLAN capability on its Ethernet interfaces with RSVP-TE control (see Section III.A), we dedicate the whole GbE bandwidth on a host's secondary NIC to a single circuit. The admission control simply checks whether the NIC is already tied up in a circuit or not.

**Data-plane configuration** on a switch consists of configuring the switch fabric to properly forward user data from an input interface to a corresponding output interface. Although the same packet forwarding function is not required at end hosts, an equivalent function is required. A CHEETAH circuit, after it is set up, is effectively a point-to-point Ethernet link between two end hosts. Typical IP network administration rules require the two ends of such an Ethernet link to have IP addresses in the same subnet. One simple mechanism to meet this requirement is to assign private IP addresses in the same subnet to the secondary NICs of all CHEETAH end hosts. However, this simple mechanism is not scalable, and given our scalability goals for CHEETAH, we chose to allow local enterprise network administrators to independently choose IP addresses for the data-plane interfaces of their CHEETAH end

hosts. This results in the two data-plane IP addresses of a CHEETAH circuit having different subnet addresses. By adding a host-specific route in the IP routing table for the distant NIC address, the IP layer at an end host can properly route datagrams onto the CHEETAH circuit.

A related problem arises from the need to map the IP address to an Ethernet (MAC) address. The destination host's data-plane MAC address is required at the source host (and vice versa) since CHEETAH circuits are essentially direct Ethernet links. The usual mechanism to obtain MAC addresses corresponding to IP addresses, called Address Resolution Protocol (ARP), is designed for local-area networks. It is a broadcast scheme, which is not suitable for wide-area networks and will incur a large delay if used in long-propagation environments, such as wide-area CHEETAH circuits. Here again, a simple update to the ARP table at the two end hosts will suffice. This explains why we obtain the IP and MAC addresses of the secondary NICs through the OCS client (see Fig. 5). The IP routing table and ARP table entries added when a circuit is setup are deleted when the circuit is released.

#### B. RSVP-TE module design and implementation

The end-host RSVP-TE module we developed for Linux hosts is based on RSVP-TE KOM [22]/DRAGON VLSR [12] code. We reused the RSVP-TE message parsing and construction functions and added software for OCS lookup, CAC and data-plane configuration. As shown in Fig. 6, the software is divided into three parts: *RSVPD*, *sig\_proc* (signaling procedures), and *bwlib*. *RSVPD* is a daemon that sends and receives RSVP-TE messages, and maintains RSVP-TE sessions. *Sig\_proc* is another daemon, which: (i) listens for circuit setup requests from local-host applications, performs connection admission control for the host's secondary NIC, and initiates RSVP-TE signaling if the request is admitted; (ii) listens for circuit setup requests passed to it by the *RSVPD* and processes them. *Sig\_proc* has a configuration file, which defines the control-plane IP address of the end host, the control port IP address of the edge switch, and the data-plane link information. *Bwlib* is a library that can be integrated into applications to enable the application to initiate the setup and teardown of circuits. For example, in our end-host RSVP-TE software package, we included an example application integrated with *Bwlib* called *circuitrequestor*, which allows users to request circuits manually by running it on the standard Linux command-line interface. This tool is useful when application source code is unavailable for integration with *Bwlib*.

*RSVPD* and *sig\_proc* are configured to automatically start up when the end host system is booted. When started, the *sig\_proc* loads configuration information from the configuration file, binds itself to a specific port, and listens to circuit setup requests generated by local applications through *bwlib*. It also registers a default session with the *RSVPD* to handle PATH messages from remote end hosts.

On the sender side, when an application requests a circuit,

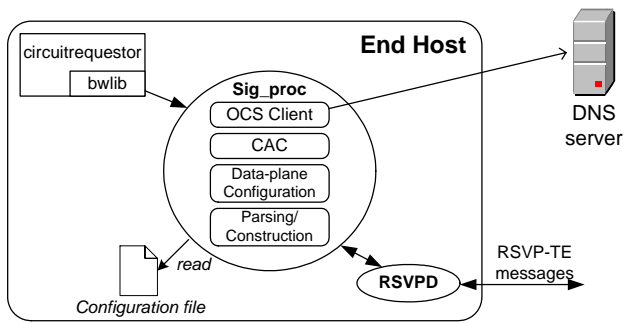


Figure 6. RSVP-TE client software architecture

*bwlib* sends the request (destination domain name and desired bandwidth) to *sig\_proc*. *Sig\_proc* then checks to see if the remote end host is on the CHEETAH network through the OCS client module, performs CAC and if successful, it forks a child process to handle this request. The newly-forked process initiates circuit setup by constructing a PATH message and asking the *RSVPD* daemon to send it to the edge switch. When the sender host receives the RESV message confirming successful circuit setup, the *sig\_proc* also updates the IP routing table and ARP table to add corresponding entries for the receiver.

On the receiver side, when a *sig\_proc* receives a PATH message from *RSVPD*, it forks a child process to handle the request. The newly-forked process creates a new session in *RSVPD* for this request and performs following steps: (i) updates the CAC table to indicate that the data-plane NIC is now reserved, (ii) sends back a RESV message to the edge switch, (iii) updates the IP routing table and the ARP table by adding entries for far-end host, and (iv) handles follow-up messages for this session.

## V. CALL SETUP DELAY MEASUREMENTS

Given that dynamic bandwidth sharing in connection-oriented switches is controlled by the signaling engine, the request handling performance of the signaling engine is critical to the scaling of networks. The faster the response times of signaling engines, the lower the cost to an application to release and re-acquire bandwidth as and when needed, and the higher the link utilization. Therefore we conducted experiments to measure circuit setup delays.

### A. Experimental setup

Measurements are taken for the circuit setup procedure between host *Zelda1* and host *Wukong* (see Fig. 2). Both *Zelda1* and *Wukong* are Dell PowerEdge servers running the Red Hat Linux operating system. As seen from Fig. 2, *Zelda1* is connected to the SOX/SLR SN16000 (*SN16k-ATL*) and *Wukong* is connected to the MCNC SN16000 (*SN16k-NC*). The two switches are interconnected by a wide-area OC-192 circuit. The RSVP-TE client software on *Zelda1* initiates the call. For this circuit, RSVP-TE PATH message processing is required at the *SN16k-ATL* and *SN16k-NC* switches, and at *Wukong*, as illustrated in Fig. 1. The primary NICs at the hosts on which signaling messages are carried and the SN16000 con-

trol ports are 100Mb/s Ethernet.

Measurements are obtained for three types of circuits: OC-1 SONET circuit, OC-3 SONET circuit, and 1Gb/s Ethernet-over-SONET (EoS) circuit. The experiment was repeated ten times for each circuit type, and the average values are reported. The variance was very small since the Internet path is lightly loaded.

Performance data collected includes end-to-end **circuit setup delay** and **per-switch signaling message processing delay**. “**Circuit setup delay**” is defined as the time interval from the instant at which an end host initiates a circuit setup to the instant at which it receives a circuit setup success confirmation from the network. In the experiment, we recorded the time difference between the instant that function *BWRequest* (the function in the *bwlib* library that is invoked by applications to request circuits) is called and the time that the function returns, which indicates that the circuit was set up successfully.

There are two types of “**per-switch signaling message processing delay**”. The first is “per-switch PATH message processing delay,” which is defined as the time interval from the instant at which an SN16000 receives a PATH message from the previous hop to the instant at which it sends out a PATH message to the next hop. Similarly we define “per-switch RESV message processing delay” as the time interval from the instant at which an SN16000 switch receives a RESV message to the instant at which it sends out a RESV message. To obtain per-switch signaling processing delays, we captured RSVP-TE messages on the control card of an SN16000 switch and recorded the time stamp of each message. We then calculated per-switch signaling message processing delays by subtracting the message arrival time from the transmit time for the corresponding outgoing message.

Besides per-switch signaling message processing delays, the end-to-end circuit setup delay consists of other two components: processing delays incurred at the two end hosts, and propagation plus emission delay of signaling messages. We also measured these delays.

### B. Experimental results and analysis

Table I shows measured data for the circuit setup procedures between host *Zelda1* and *Wukong*. The first column shows the circuit type. The second column shows the end-to-end circuit setup delay. The third and the fourth columns show the per-switch signaling message processing delays observed at the *SN16k-NC* for the PATH and RESV messages, respectively. Finally, we shown the round-trip signaling message propagation plus emission delays.

We can see that end-to-end circuit setup delays for an OC-1 circuit and an OC-3 circuit are both around 166ms. The reason that these two delay values are similar is that the procedures for setting up these two pure-SONET circuits are the same.

On the other hand, the end-to-end circuit setup delay for a 1Gb/s EoS circuit is much higher (around 5s). This is because

TABLE I SIGNALING DELAYS INCURRED IN SETTING UP A CIRCUIT BETWEEN ZELDA1 AND WUKONG ACROSS THE CHEETAH NETWORK

Circuit type	End-to-End circuit setup delay (s)	Processing delay for PATH message (s) at the <i>SN16k-NC</i>	Processing delay for RESV message (s) at the <i>SN16k-NC</i>
OC-1	0.166103	0.091119	0.008689
OC-3	0.165450	0.090852	0.008650
1Gb/s EoS	4.982071	4.907113	0.008697
Round-trip signaling message emission and propagation delays between <i>SN16k-ATL</i> and <i>SN16k-NC</i> : 0.025s			

21 OC-1 circuits are established one after another between the two switches before the Ethernet segments are mapped to this 21-OC1 circuit. It is purely an artifact of the current release of the SN16000 software whereby virtually concatenated signals, while supported in the data plane are not yet supported in the control-plane. We expect to receive a software upgrade that would reduce EoS circuit setup delays to the same order of magnitude as pure-SONET circuits.

The PATH message processing delay measured at the *SN16k-NC* for the 1Gb/s EoS circuit can be validated by summing 42 per-switch PATH message processing delays (one on each switch for each OC-1), 21 times round-trip propagation delay, and 42 signaling message emission delays between *SN16k-ATL* and *SN16k-NC*. For example, since the per-switch signaling processing delay for a pure-SONET circuit is 0.091s and the summation of round-trip signaling message propagation and emission delay is 0.025s as shown in Table I, an approximation of PATH message processing delay for a 1Gb/s EoS circuit is  $42 \times 0.091 + 21 \times 0.025 = 4.35$ s. Roughly speaking, this explains the 4.9s PATH message processing delay seen at the *SN16k-NC* for the 1Gb/s EoS circuit.

From these results we conclude that the signaling message processing delay dominates the end-to-end circuit setup delay. Any enhancements to signaling message processing speeds would be useful to reduce circuit setup delays.

## VI. CONCLUSIONS

The CHEETAH experimental network aims at creating a scalable connection-oriented network for a wide range of applications. We designed and deployed the CHEETAH experimental network, which consists of Sycamore SN16000 switches, Linux PCs equipped with the end-host CHEETAH software, and wide-area OC-192 circuits. We developed a CHEETAH end-host software package, which enables end hosts to dynamically initiate and accept circuit setup requests. Wide-area signaling experiments conducted on the CHEETAH network show that the end-to-end circuit setup delay across a 25ms round-trip propagation-delay path is around 166ms.

The CHEETAH implementation demonstrates that a solu-

tion of extending dedicated circuits to end hosts and enabling end-host initiated signaling is feasible as GMPLS control-plane protocols are quite mature today. Security concerns with extending control-plane connectivity to end hosts can be effectively solved with minimal security hardware and software.

Our original design of requiring end hosts to be directly connected to the Ethernet-to-SONET circuit gateways is expensive. A more feasible design is to use Ethernet switches within enterprises and configure VLANs across these switches to create virtual circuits, which are then mapped on to SONET circuits. More generally, the CHEETAH solution of creating end-to-end Ethernet-EoS-Ethernet circuits should be extended to heterogeneous connections, with VLAN, MPLS, SONET and WDM segments.

## ACKNOWLEDGEMENTS

We thank colleagues from NCSU, ORNL, MCNC, NLR, GaTech, SLR, the NSF Dragon project, and Sycamore Networks, for their help in creating the CHEETAH experimental network. We specially thank Vijay Pandian from Sycamore Networks for his valuable comments on this paper.

## REFERENCES

- [1] M. Veeraraghavan, X. Zheng, H. Lee, M. Gardner, and W. Feng, "CHEETAH: Circuit-switched High-speed End-to-End Transport Architecture," *Proc. of Opticomm 2003*, Dallas, TX, Oct. 13-17, 2003.
- [2] E. Mannie, "Generalized Multi-Protocol Label Switch (GMPLS) Architecture," *IETF RFC 3945*, Oct. 2004.
- [3] L. Berger, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions," *IETF RFC 3473*, Jan. 2003.
- [4] D. Katz, K. Kompella, and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2," *IETF RFC 3630*, September 2003.
- [5] J. Lang, "Link Management Protocol (LMP)," *IETF RFC 4204*, October 2005.
- [6] B. Metcalfe, "Metcalfe's Law: A network becomes more valuable as it reaches more users," *Infoworld*, Oct. 2, 1995.
- [7] "CANARIE's CA\*net 4," <http://www.canarie.ca/canet4/index.html>.
- [8] "OMNInet," <http://www.icaire.org/omninet/>.
- [9] "SURFnet," <http://www.surfnet.nl/info/en/home.jsp>.
- [10] "UKLight," <http://www.uklight.ac.uk/>.
- [11] "UltraScience net," <http://www.csm.ornl.gov/ultranet/>.
- [12] "Dynamic Resource Allocation via GMPLS Optical Networks (DRAGON)," <http://dragon.east.isi.edu/>.
- [13] "User-controlled LightPaths (UCLP)," <http://www.canarie.ca/canet4/uclp/>.
- [14] "Generic Framing Procedure (GFP)," ITU-T Recommendation G.7041, October 2001.
- [15] American National Standards Institute, "Synchronous Optical Network (SONET) - Payload Mappings," ANSI T1.105.02, 1998.
- [16] "NetScreen-5XT," [http://www.juniper.net/products/integrated/ns\\_5series.html](http://www.juniper.net/products/integrated/ns_5series.html).
- [17] "Openswan," <http://www.openswan.org/>.
- [18] "GridFTP," <http://www-unix.globus.org/toolkit/docs/3.2/gridftp/>.
- [19] A. P. Mudambi, X. Zheng, M. Veeraraghavan, "A Transport Protocol for dedicated end-to-end circuits," *Proc. of IEEE ICC 2006*, June 11-15, 2006, Istanbul, Turkey.
- [20] X. Zheng, et al., "CHEETAH end-host software design document," <http://cheetah.cs.virginia.edu/software/cheetah-software.pdf>.
- [21] G. Swallow, et al., "Generalize Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model," *IETF draft*, Oct. 2004.
- [22] Ralf Steinmetz, et al., "KOM RSVP Engine," <http://www.kom.tu-darmstadt.de/en/downloads/software/kom-rsvp-engine/>.