

Control Plane for Advance Bandwidth Scheduling in Ultra High-Speed Networks

Nageswara S. V. Rao, Qishi Wu,
Steven M. Carter, William R. Wing
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831, USA

Amitabha Banerjee, Dipak Ghosal,
Biswanath Mukherjee
Department of Computer Science
University of California
Davis, CA 95616, USA

Abstract—A control-plane architecture for supporting advance reservation of dedicated bandwidth channels on a switched network infrastructure is described including the front-end web interface, user and token management scheme, bandwidth scheduler, and signaling daemon. A path computation algorithm for bandwidth scheduling is proposed based on an extension of Bellman-Ford algorithm to an algebraic structure on sequences of disjoint non-negative real intervals. An implementation of this architecture for UltraScience Net is briefly described.

I. INTRODUCTION

An increasing number of large-scale science and commercial applications are producing colossal amounts of data, on the order of terabytes currently and petabytes in the near future. Since the data providers and consumers in these applications are often geographically distributed over a wide-area environment, ultra high-speed dedicated connections are needed to transfer the data to remote sites for a variety of purposes including data mining, data consolidation and alignment, storage, visualization, and analysis. For example, large simulation datasets produced by an eScience application on a supercomputer may be archived at a remote storage site, or bank transaction records or inventories of a large chain of departmental stores may be synchronized during off-peak hours. Dedicated high bandwidth channels are critical in these applications to ensure timely task completion, which in turn necessitates a high-performance control plane capable of scheduling such channels in advance.

We propose a generalized control plane to support in-advance reservation of dedicated channels over ultra high-speed networks. The proposed control plane framework shown in Figure 1 consists of the following components: (a) client interface, (b) server front-end, (c) user management, (d) token management, (e) database management, (f) bandwidth scheduler, and (g) signaling daemon. The interactions between these components take place either over the data plane or control plane to accomplish the tasks of user-specified bandwidth reservation, path computation and network signaling.

II. CONTROL-PLANE COMPONENTS

Depending on the system configuration, the control plane operations can be coordinated by a central management node or by a set of nodes distributed over a network. A user can

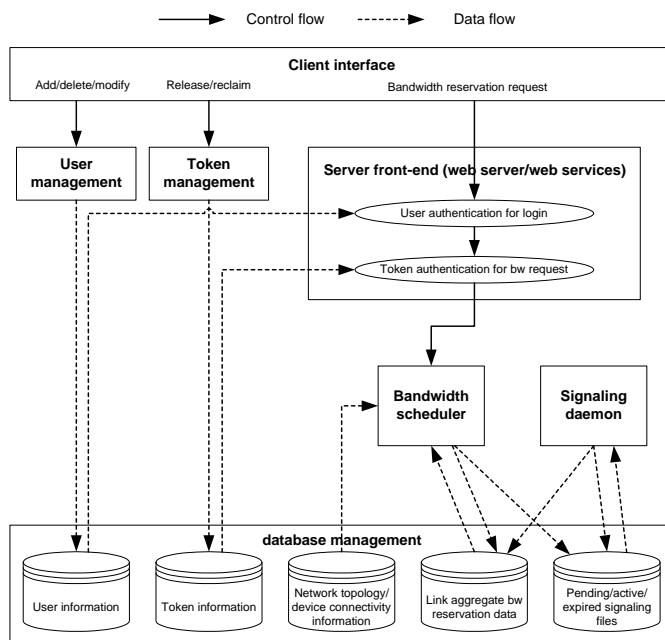


Fig. 1. Framework of control plane: function components, control flow, and data flow.

remotely interact with the system through a web browser, or a web client, for example, using SOAP (Simple Object Access Protocol)-based XML message exchange. Accordingly, the server front-end could deploy a web server or web service that accepts bandwidth reservation requests from users with valid credentials. The user management module supports a special group of users with administrative privileges to add, delete or modify user account information.

User sites are connected through their assigned ports on the edge switches of the network infrastructure. A token-based scheme is used for authorization and coordination of channel setup. Multiple tokens are provided to users for their assigned ports, which they can release to other users and reclaim them as needed. A channel reservation request is honored only if the tokens at both ends are either owned by or released to the user making the request. It is implicitly assumed that users at both ends will work out their connectivity mechanisms and

policies before the tokens are released.

Based on the network topology information and existing link bandwidth allocations, the scheduler computes an appropriate path in response to a specific user request. Upon path computation, a signaling record is generated and the bandwidth allocations of each link along that path are updated accordingly. The signaling daemon periodically examines active or expiring signaling records as shown in Figure 2. For each active or expiring signaling record, the daemon invokes appropriate signaling scripts to set up or tear down the connections along the computed or established path, respectively. The aggregate bandwidth reservation data for each component link is updated if the signaling actions are successful. Note that the time interval at which the signaling daemon is periodically activated must be chosen to be compatible with the finest resolution of the bandwidth reservation time.

III. BANDWIDTH SCHEDULING

A bandwidth scheduler computes the channel bandwidth allocations with flexible user options on time windows and bandwidth requirements. A user can request a layer-1 or layer-2 dedicated channel at current time (i.e. on-demand) or in a future time slot. The bandwidth scheduling algorithms are based on extending the classical shortest path computations using a closed semiring algebraic structure defined on sequences of disjoint intervals on real line. Appropriate paths may be computed to meet the user needs for: (i) a specified bandwidth in a specified time slot, (ii) earliest available time with a specified bandwidth and duration, (iii) highest available bandwidth in a specified time slot, and (iv) all available time slots with a specified bandwidth and duration. All these algorithms are mathematically guaranteed to return the correct answers. The first three algorithms are straight forward extensions of the classical Dijkstra's algorithm [3]. The last algorithm is based on an extension of Bellman-Ford algorithm as will be described next; this algorithm is an improvement over the transitive closure algorithm described in [4] in terms of time complexity.

A network is represented as graph $G = (V, E)$ where each node represents a switch and each edge represents a link. For each edge $e \in E$, we have a list of bandwidth reservations specified as a piecewise constant function of time. Algorithm ALL-SLOTS lists all time-slots during which a path with bandwidth b is available for duration t from node s to node d . For each $e \in E$, we generate a list L_e of disjoint intervals such that bandwidth b is available on e for duration t starting any time within any interval. The algorithm is essentially the well-known Bellman-Ford algorithm [3] with the modification to utilize operations \oplus and \otimes that correspond to point-wise merging and intersection of intervals of the corresponding lists, respectively. We have $L_e \oplus \{\phi\} = L_e$, $L_e \oplus \{\mathbb{R}^+\} = \{\mathbb{R}^+\}$, $L_e \otimes \{\mathbb{R}^+\} = L_e$, and $L_e \otimes \{\phi\} = \{\phi\}$, where ϕ is the empty interval (set) and \mathbb{R}^+ is the infinite interval (set) of non-negative reals.

algorithm ALL-SLOTS;

1. $\mathcal{D}(s) \leftarrow \{\mathbb{R}^+\}$;
 2. $\mathcal{D}(v) \leftarrow \{\phi\}$ for all $v \neq s$;
 3. **for** $k = 1, 2, \dots, n - 1$ **do**
 4. **for** each edge $e = (v, w)$ **do**
 5. $\mathcal{D}(w) \leftarrow \mathcal{D}(w) \oplus \{\mathcal{D}(v) \otimes L_e\}$;
 6. **return**($\mathcal{D}(d)$).
-

We will now show that this algorithm maintains the following invariant: at the end of iteration k , $\mathcal{D}(v)$ contains the intervals corresponding to all starting times of all paths of length at most k from s to v with bandwidth b for duration t . Consider a path of length at most $l \leq k$ edges e_1, e_2, \dots, e_l listed in order from s to v . Consider $i_1 < i_2 < \dots < i_l$ to be the iterations during which the edges e_1, e_2, \dots, e_l were selected, respectively; such sequence exists since every edge is considered in every iteration. Since these edges are considered sequentially in time and $\mathcal{D}(s) \leftarrow \{\mathbb{R}^+\}$ initially, we have $\bigotimes_{i=1}^l L_{e_i} \subseteq \mathcal{D}(v)$, which ensures that all starting times at which this path provides bandwidth b for duration t will be contained in $\mathcal{D}(v)$. By the monotonicity of \oplus operation, once starting times are placed on $\mathcal{D}(v)$, they will not be removed. By initialization $\mathcal{D}(v) \leftarrow \{\phi\}$ for $v \neq s$ and Step 5, only the starting times corresponding a path providing bandwidth b for duration t will be added to $\mathcal{D}(v)$.

The complexity of this algorithm is polynomial, namely $O(nm)$, in terms of \oplus and \otimes operations. For a sparse graph $m < n^2$, which is the case for wide-area dedicated networks, $O(nm)$ is tighter than $O(n^3)$ of transitive-closure algorithm of [4].

To our knowledge, UltraScience Net (USN) [4] has the first implementation of a control plane with mathematically-validated advance scheduling capability. Such advance scheduling is not a part of MPLS (Multiple Protocol Label Switching) and GMPLS (Generalized MPLS) specification [5]. Consequently, this capability is not currently supported by GMPLS-based networks such as CHEETAH [6] and DRAGON [1]. While the advance reservation is supported over IP networks using MPLS by OSCARS [2] of ESnet, their underlying path computation algorithm does not explore all paths with available bandwidth inside the network.

IV. ULTRASCIENCE NET

UltraScience Net is a wide-area experimental network testbed to support the development of networking capabilities needed for next-generation computational science applications [4]. USN provides dedicated high-bandwidth channels for large data transfers, and also high-resolution, high-precision channels for fine control operations. The data plane of USN consists of several thousand miles of dual OC192 connections, which can be provisioned at layer-1 and layer-2 at OC-1 and STS-1 resolutions, respectively. The control plane integrates a number of functional modules including bandwidth reservation, scheduling, signaling, token-based authorization and web-based user/client interface. As shown in Fig. 3, the

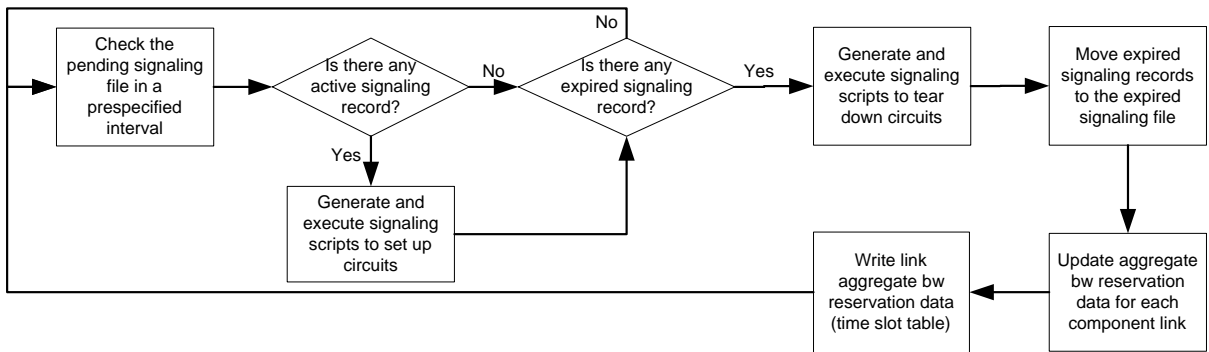


Fig. 2. Control structure of signaling daemon.

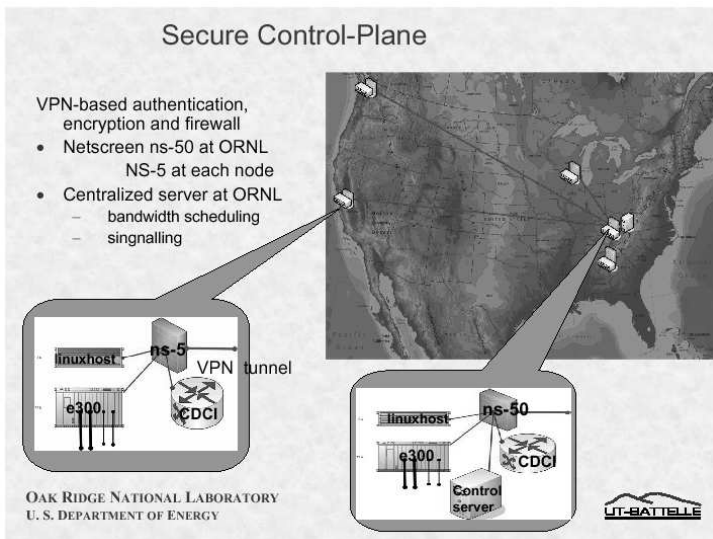


Fig. 3. UltraScience Net control plane.

authenticated and encrypted control traffic is carried over out-of-band Virtual Private Network (VPN) tunnels implemented over shared IP network using NetScreen devices. We provide below a brief coverage of the control plane implementation of USN including on-going and future efforts.

User sites are connected to USN via their assigned ports on the edge or core switches. A user can request a dedicated path between two USN ports using the token-based authorization system. The VLAN (Virtual Local Area Network) tags are carried transparently across layer-2 USN channels, and hence VLANs, if employed, must be appropriately coordinated at both end points. After logging into the system with valid credentials, users can utilize a web-based graphical interface to make their bandwidth reservation requests and check the current allocations on various links. A web server is currently being developed to support web-clients based on WDSL (Web Service Description Language) description and SOAP-based communication of XML messages. The switches in the core, Ciena CDCIs, are signaled using TL1 commands, and switches

at the edge, Force10 E300s, are signaled using CLI commands. In both cases, EXPECT scripts are utilized by the signaling daemon to log into the switches via encrypted VPN tunnels to issue the commands. Our future plans include GMPLS wrappers to the USN control plane to facilitate peering with other networks such as NSF CHEETAH network.

The current version of USN control plane is implemented in C++ using CGI and PHP scripts for the server and JavaScript and HTML for user interface. It is deployed on a central management node on a Linux workstation at Oak Ridge National Laboratory. Data structures used to represent user accounts, network topology, device configurations and aggregate link bandwidth reservations are stored and maintained in their underlying databases. These databases are updated automatically when scheduling or signaling activities take place. Also, the bandwidth allocations of each link in the network are displayed in real time to reflect the utilization of system resources.

ACKNOWLEDGMENT

This research is sponsored by the High Performance Networking Program of the Office of Science, U.S. Department of Energy, under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC, the Defense Advanced Projects Research Agency under MIPR No. K153, and by National Science Foundation Under Grants No. ANI-0229969 and No. ANI-0335185.

REFERENCES

- [1] Dynamic resource allocation via GMPLS optical networks. <http://dragon.maxgigapop.net>.
- [2] On-demand secure circuits and advance reservation system. <http://www.es.net/oscars>.
- [3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill Book Co., New York, 1990.
- [4] N. S. V. Rao, W. R. Wing, , S. M. Carter, and Q. Wu. Ultrascience net: Network testbed for large-scale science applications. *IEEE Communications Magazine*, 2005. in press, expanded version available at www.csm.ornl.gov/ultranet.
- [5] N. Yamanaka, K. Shiimoto, and E. Oki. *GMPLS Technologies*. CRC Taylor Francis Pub, 2006.
- [6] X. Zheng, M. Veeraraghavan, N. S. V. Rao, Q. Wu, and M. Zhu. CHEETAH: Circuit-switched high-speed end-to-end transport architecture testbed. *IEEE Communications Magazine*, 2005.