DRAGON Software Suite

Release 1.0

1 Introduction

The DRAGON project is developing technology and deploying network infrastructure to allow dynamic provisioning of deterministic network paths in direct response to end-user requests. The goals include interdoman provisioning across heterogeneous network technologies while including mechanisms for AAA (Authentication, Authorization, and Accounting) and scheduling. A detailed description of the DRAGON Architecture is not presented here, but is available via the DRAGON Architecture document [1]. A reference implementation of this architecture is being constructed in the Washington D.C. area in the form of the DRAGON Network [2].

The initial motivation for these advanced network services in support of the eScience community who desire the dynamic acquisition of dedicated and deterministic network resources to link expensive equipment such as radio telescopes, computational clusters, storage arrays, visualization facilities, remote sensors, and other instruments into globally distributed and application specific topologies. It is expected that other communities and applications will also be able interested in and able to utilize these types of services as well.

The purpose of this document is to describe the DRAGON Software Suite, release 1.0. The DRAGON software suite is a collection of software to enable dynamic provisioning of network services. This document includes description of the functional elements, software components, installation instructions, and configuration guides. A usage example is also provided to facilitate description of how components interact during the course of a dynamically provisioned service.

This software release does not include all the features and capabilities as described in [1]. While this document may mention some future capabilities to facilitate description of the current feature set, in general only those features and capabilities present in this release are described herein.

The DRAGON architecture utilizes Generalized MultiProtocol Label Switching (GMPLS) as the basic building block for network element control and provisioning. As such, descriptions of the functional elements and software components utilizes GMPLS terminology. Additional descriptions regarding GMPLS and it application to DRAGON are available in reference [1].

2 DRAGON Functional Elements

There are several functional elements provided by this version of the DRAGON Software Suite:

- Network Aware Resource Broker (NARB)
- Virtual Label Switch Router (VLSR)
- End-System Agent (ESA)
- Application Specific Topology Builder (ASTB)
- Constrained Shortest Path First (CSPF) Module
- GMPLS NUTTCP (G-NUTTCP)

These are described in the sections that follow.

2.1 NARB

The NARB is an agent which represents a local Autonomous Domain (AD). The NARB serves as path computation engine from which end-systems or other devices can query to find out about availability of traffic engineered paths between specified source and destination pairs. The NARB is also responsible for inter-domain routing. NARBs peer across domains and exchange topology information to enable inter-domain path computation and Label Switched Path (LSP) provisioning. This inter-domain topology exchange can be based on the actual topology as discovered by listening to the local Open Shortest Path First (OSPF) protocol, or optionally based on an "abstracted" view of the domain topology (generated by configuration file or automatic synthesis of the OSPF link state database).

The specific NARB capabilities in this release include:

- intra-domain routing listener
- intra-domain path computation
- inter-domain routing
- domain level topology abstraction (in support of inter-domain routing)
- inter-domain path computation
- Label Switched Path (LSP) management

2.2 V LSR

The VLSR provides a mechanism to integrate non GMPLS equipment and network regions into the end-to-end GMPLS provisioned services. The VLSR translates standard GMPLS protocols into device specific protocols, to allow dynamic reconfiguration of non-GMPLS aware devices. The combination of a PC which runs the GMPLS based control plane software and the switch fabric is referred to as a VLSR. In this release the underlying switch fabric is a RFC 2674 [3] compliant ethernet switch. The architecture is such that different types of switch fabrics can readily be added. In addition methods other than RFC 2674 control could also be added. The GMPLS based control software functionality includes OSPF with GMPLS Traffic Engineering (OPSF-TE) [4] extensions and Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) [5] extensions.

The following ethernet switches have been tested with this release of the VLSR.

Dell, x series, xxx software release
Extreme Summit i series, 7.0 software release
Foundry, x series, x software release

2.3 ESA

The ESA is software that runs on the end-systems which terminate the data plane (traffic engineering) link of the provisioned service. This is the software that participates in the GMPLS protocols to allow for on-demand end-to-end provisioning from end-system to end-system. There is also an ESA API which is utilized by the ASTB in order to translate user requests into protocol provisioning actions.

The DRAGON architecture identifies four distinct types of end-systems as follows:

A. Peer-to-Peer End-System (ES)
   - ES terminates the data plane link
   - ES terminates the control plane link
   - ES has both RSVP and OSPF

B. Dumb ES, Peer-to-Peer Proxy
   - ES terminates the data plane link
   - Proxy terminates the control plane link
   - Proxy has both RSVP and OSPF

C. Overlay ES
   - ES terminates the data plane link
   - ES terminates the control plane link
   - ES use UNI (RSVP based) Signaling

D. Overlay Proxy ES (Third-Party Signaled)
   - ES terminates the data plane link
   - Proxy uses UNI (RSVP based) signaling on behalf of ES

This release of the DRAGON Software Suite includes the functionality for above ES types A and B. In this context, ES is a very broad term. It generally means any device which finds it self on the edge of a DRAGON enabled dynamically provisioned network. This could include a host, a computational cluster, a router, a radio telescope, and various other networked devices.

2.4 ASTB

ASTB provides an API which is utilized by applications to request network services. Applications can use this API to request a dynamic path set up across an DRAGON enabled network. This API includes a simple message set to allow an application to request a dedicated network path between a specified source/destination pair identified by IPv4 addresses. In future releases, the ASTB will provide a more general "topology builder" functionality. This will allow an application to request a topology to be built which consists of multiple point to point LSPs in a user specified configuration. This release includes the first building block of that larger vision by allowing a single LSP specification per request.

The interface to the ASTB API is via ????(C, C++, XML, other). The application may be another computer program or a web browser driven by human interaction. This release of the DRAGON Software Suite includes both the standard message set API as well as a web browser control interface. The ASTB also utilizes the ESA API to translate user requests into end-system protocol provisioning actions.

2.5 CSPF Module

The CSPF functionality provided in this release provides the ability to compute traffic engineered paths based on the OSPF-TE derived Link State DataBase (LSDB). This software is integrated into the open source routing suite software as described in the following section and understands how to interpret those LSDB data structures. In addition, the CSPF software is provided as a standalone module and includes an API to allow use elsewhere.

The current CSPF implementation is limited to single LSP regions and considers the standard GMPLS-TE constraints of bandwidth availability and interface switching capability. This will be expanded in future releases to allow for multi-regions LSP path calculation and include additional constraints such as network element adaptation capabilities and physical impairments for all-optical paths.

2.6 G-NUTTCP

G-NUTTCP is a modified version of NUTTCP[6]. This modification integrates the ASTB API functionality to initiate the provisioning of and end-to-end deterministic path, and immediately run a performance test to verify network path performance. This is a convenient tool to test end-to-end performance for the purposes of network debugging or as a prior step before other applications use the network path.

3 DRAGON Software Suite Components

The DRAGON Software Suite is provided as a single tar file and is available here:

- http://dragon.maxgigapop.net/dragonsoftware/dragon-v1.0.tar.gz

This tar file contains the following component tar files:

- dragon-v1.0-gnu-zebra.tar.gz
- dragon-v1.0-gnu-rsvp.tar.gz
- dragon-v1.0-cspf.tar.gz
- dragon-v1.0-astb.tar.gz
- nuttcp-vx.x.tar.gz

These components are described below:

DRAGON Zebra (dragon-v1.0-gnu-zebra.tar.gz)
The DRAGON project has extended the open source GNU Zebra [7] routing software package to include required GMPLS functionality. The GNU Zebra distribution is a routing protocol suite and includes multiple network protocols such as RIP, OSPF, BGP and others. The DRAGON modifications included the following:

- Addition of GMPLS TE extensions for OSPF
- Addition of capability to act as and Label Switch Router (LSR) responsible for an ethernet switch fabric.
- Addition of a DRAGON specific Command Line Interface (CLI) for LSP instantiation and control
- Addition of NARB Functionality and associated CLI
- Incorporation of CSPF Routing Module
- Addition of XML interface for CLI control

DRAGON KOM-RSVP (dragon-v1.0-gnu-rsvp.tar.gz)
The DRAGON project has extended the open source KOM RSVP Engine [8] from The Technical University of Darmstadt to include required GMPLS functionality. The KOM RSVP Engine provides an implementation of RSVP. The DRAGON modifications included the following:

- Addition of GMPLS Traffic Engineering Extensions for RSVP
- Extension of the RSVP API to allow control by DRAGON Zebra CLI and the DRAGON ASTB component
- Addition of functionality to allow control of RFC 2674 compliant ethernet switches

DRAGON ASTB (dragon-v1.0-astb.tar.gz)
The ASTB provides the functionality as described in Section 2.4.

DRAGON CSPF (dragon-v1.0-cspf.tar.gz)
The CSPF module is provided as a separate module which includes an API in the form of function calls and provides the functionality as described in Section 2.5. This module is also integrated into the DRAGON Zebra distribution described above.

G-NUTTCP (nuttcp-vx.x.tar.gz)
The G-NUTTCP provides the functionality as described in Section 2.6.

4 Documentation

This section provides instructions for the install and configuration of the DRAGON Software Suite version 1.0. Configuration parameters are dependent on functionality desired, however once the entire release is installed all necessary software components will be available.

NOTE: Sections 4.1 thru 4.7 are incomplete but the idea is that these would be the same text files that would be included as READMEs in the tarballs.

4.1 DRAGON Software Suite Install Guide

The system requirements for install and compilation of this release are as follows:

- FreeBSD (x.x or newer release) or LINUX (2.x kernel or newer)
- gcc version x.x or newer
- Net-SNMP Version xx installed

The below steps should allow one to complete the install and compile this software release.

1. Download the DRAGON Software Suite from:

http://dragon.maxgigapop.net/dragonsoftware/dragon-v1.0.tar.gz

2. Install software package in desired directory:

```
gzip -cd dragon-v1.0.tar.gz  | tar xvf -
```

3. Install dragon-v1.0-gnu-zebra.tar.gz component

```
gzip -cd dragon-v1.0-gnu-zebra.tar.gz  | tar xvf -
```

4. Compile dragon-v1.0-gnu-zebra

```
cd dragon-gnu-zebra
./configure --enable-dragon CFLAG=-g CPPFLAG=-g
 make
```

5. Install dragon-v1.0-gnu-rsvp.tar.gz

```
gzip -cd dragon-v1.0-gnu-rsvp.tar.gz  | tar xvf -
```

6.Compile dragon-v1.0-gnu-rsvp

```
cd dragon-kom-rsvp
./configure --with-snmp=/usr/local CFLAG=-g CPPFLAG=-g
gmake clean
gmake depend
gmake
gmake install
```

etc.....

4.2 NARB Configuration Guide

4.3 VLSR Configuration Guide

4.4 ESA Configuration Guide

4.5 ASTB Configuration Guide

4.6 CSPF Configuration Guide

4.7 G-NUTTCP Configuration Guide

5 Usage Example

NOTE: This section would include a drawing of an actual network configuration complete with IP addresses, showing DRAGON functional elements, with arrows to explain message flows between elements

REFERENCES

1 DRAGON Team, Dragon Architecture Description, May 2005

2 DRAGON Team, DRAGON Network Description, May 2005

3 E.Bell, A.Smith, P.Langille, A.Rijhsinghani, K.McLoghrie, "Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering and Virtual LAN Extenstions", RFC 2674, August 1999.

4 K. Kompella, Y. Rekhter, "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching", draft-ietf-ccamp-ospf-gmpls-extensions-12.txt, April 2004.

5 Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

6 Fink, B, NUTTCP, www.xxxxx

7 GNU Zebra Routing Protocol Suite, www.zebra.org

8 KOM RSVP Engine, Technical University of Darmstadt, www.kom.e-technik.tu-darmstadt.de/rsvp/