# Control Plane for Carrier-Grade Ethernet Network

Olli-Pekka Lamminen, Marko Luoma, Jukka Nousiainen, and Taneli Taira
Department of Communications and Networking
Helsinki University of Technology
P.O. Box 3000, 02015 TKK, Finland
E-mail: {firstname.lastname}@tkk.fi

***Abstract*—We present a control plane architecture for a service-oriented carrier-grade Ethernet transport network. Our control plane is based on a centralized Management System and separate Control Elements. Network topology information is gathered from the Control Elements with Ethernet-extended IS-IS protocol. Based on the topology, the Management System is able to provision transport tunnels in the domain in an optimal way, utilizing a dedicated path computation element. RSVP-TEEth protocol, an extension of RSVP-TE, is used to signal the tunnels in the network. The control plane is developed as a part of a more complete system, which includes forwarding plane realized with network processors and a management plane for provisioning both intra-domain services, and inter-domain services between multiple domains.**

## I. Introduction

THE nature of networks is continuously changing. We have seen the tremendous growth of Internet users during the last decade [1], and the move from wired to wireless subscriptions worldwide [2], [3]. At the same time it has become essential for large businesses to be connected across the globe and for businesses of any size to have a strong visibility in and high quality access to the Internet [4].

Meanwhile the network operators have been moving their services, be it legacy voice communications or packet data transport, to a unified backbone network [5]. This helps to lower the operational expenditures as the same system can be used to deliver a multitude of services. This also creates a need for new systems capable of delivering different types of services simultaneously in an easily maintainable and operable manner.

Currently many of the core networks are running IP over MPLS over different link layer protocols like Ethernet or SONET/SDH. On the end user side the dominating networking technology is IP over Ethernet. The common denominator here is IP, which itself is under constant pressure from e.g. expected address exhaustion [6] and disregard of its original design principles in modern ISP networks [7]. Instead of IP we want to lower the common denominator to Ethernet, which we think can support service oriented networking and carrier-grade traffic engineering [8].

This work was partly supported by the EU funded FP7 strep project ETNA Ethernet Transport Networks, Architectures of Networking (project nr 215462) [9].

Manuscript received July 30, 2009.

At the moment there are many technologies competing to be the dominant carrier-grade Ethernet Transport technology, MPLS-TP and 802.1Qay being the most prominent ones. An important issue with all Ethernet Transport solutions is how the control plane is realized – with static provisioning by an NMS/OSS, some sort of dynamic control mechanism, or a hybrid of both.

In this paper we describe our implementation of a carrier-grade Ethernet control plane developed as a part of EU FP7 funded ETNA project. Our control plane implementation realizes the service oriented transport network architecture developed in ETNA [9], and allows versatile control over the transport and transport service layers of that architecture. Getting rid of IP in the transport and focusing purely on extending Ethernet's capabilities enables traffic management on a single layer greatly reducing the complexity and compatibility issues of multi-layer management [10]. With MAC-in-MAC encapsulation the network can also natively support L2 VPNs and other more advanced services. The separation of core and customer addresses protects the operators network from outside intervention and makes the core transparent to the end user. These ideas have previously been demonstrated on proof of concept level in [11], but this is the first larger scale implementation of this type of network architecture in practivce we are aware of.

The rest of the paper is organized as follows. In section II we introduce the architetural overview of our implementation. We further describe the essential parts of our design, the Management System in section II-A and the Control Elements in section II-B. In section III we show how the different parts of the system function together in order to realize a simple point to point connection between two clients. In section IV we give our conclusions and outline future work planned to further refine our system. The control plane implementation will be demonstrated together with forwarding plane from BGU during the Fall 2009 in both intra- and inter-domain environments.

## II. Control Plane Architecture

Our control plane is designed to support ETNA's three-layer architecture described in [12]. In this architecture the network is divided into transport layer, transport services layer, and network services layer. The transport layer handles packet transmission through the network. Transport services layer

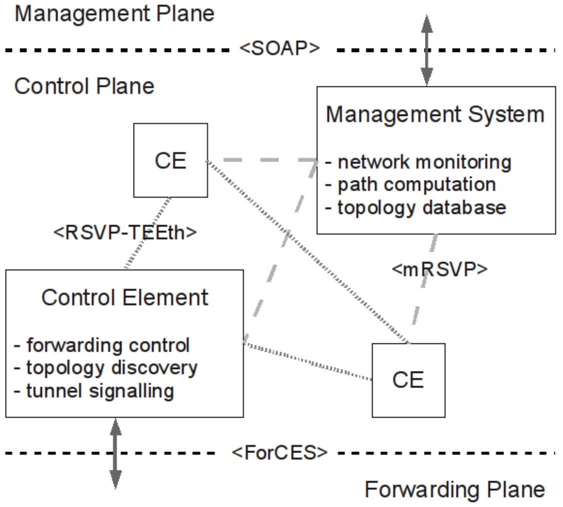


Fig. 1. Control plane architecture

adds a layer of services on top of this, supporting e.g. point-to-point and point-to-multipoint tunnels and virtual circuits. Network services layer adds even more refined network functionalities, like identities and mobility, which can be enabled on an on-demand basis for customers. Our control plane implementation provides for transport and transport services layer.

Transport level primitives our control plane supports are called tunnels. In our architecture a tunnel is a connection between two or more network edge nodes, to which customers can then be attached. This means that a single tunnel can transport more than one customer's traffic. The separation of customers' traffic is done at the tunnel end points and the customers' identities play no role during the transport itself.

Tunnels have multiple parameters which define the tunnel's guaranteed and maximum bandwidth, reserved capacity, and participation in forwarding. Tunnels are further assigned to service envelopes, which define additional criterion like delay and jitter bounds, OAM monitoring, and protection options. This allows the network operator to define service classes which can be offered to customers, e.g. high-availability VPNs with protection, or low-latency connections for real-time applications.

Traffic inside the core network uses custom Ethernet framing resembling IEEE-802.1ah, which allows us to tag the traffic inside the network with tunnel, service, and customer identifiers, as well as preserve the original frame of the customer network as payload. Customizable frame type definitions stored in XML format supported by the forwarding plane hardware allow us to support multiple different customer framings at the same time, and translate them to the core frame format and back on the fly at the ingress/egress points [13].

Internally the control plane is divided into management and control functionalities as shown in figure 1. Management functionalities, like path computation and network monitoring, are centralized in a Management System described in section II-A. Control functionalities, like signalling tunnels and directing the hardware responsible traffic forwarding, are performed in Control Elements (CE) distributed in the network and detailed in section II-B.

We have designed the different systems to communicate with each other with common, well known technologies to ease the development and allow for future research and development with our systems by interested 3rd parties as well. The Management System is connected to external management plane via SOAP interface. Control Elements are connected to data plane with ForCES protocol, which is also where the term 'Control Element' is adopted from.

## A. *Management System*

In our implementation, the control plane is controlled by a single Management System (MS). The MS is responsible for directing Control Elements (CE), maintaining an image of network topology, calculating paths for tunnels in the network, recording and resolving fault situations, and monitoring and displaying network state and health.

Centralized architecture for the Management System was selected mainly because of easier development for the initial prototype. Centralized management allows us to avoid additional development costs of an intra-management synchronization protocol. Coordinated path computation is also essential for our architecture to achieve carrier-grade quality and performance [14]. The architecture itself does not limit us to this approach only.

Management System has two primary communications interfaces. First interface is between the MS and Control Elements, and thus is internal to the control plane. The second interface is located between the control plane and an external

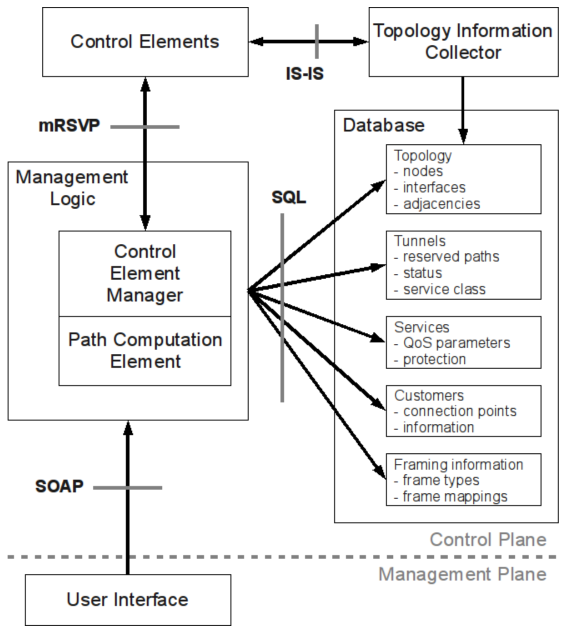


Fig. 2. Management System components

management plane, and is used for network monitoring and controlling purposes.

The MS uses an RSVP-like protocol (mRSVP) to instruct, update, and monitor CEs. An RSVP-like protocol was selected for this purpose because the control messages between CEs are sent with RSVP-TE with Ethernet extensions (RSVP-TEEth), and this allows us to use the same basic message library for both communication interfaces. Also, as we aim for pure Ethernet based solution, RSVP does not rely heavily on existing IP layer as e.g. SNMP does.

For communications between the MS and external management plane (e.g. 3rd party OSS / NMS) we are using SOAP. SOAP, as a well established standard for implementing e.g. web services, is a good solution for enabling fast development of a management interface, and also allows for more flexible integration of possible 3rd party developed systems to manage our control plane. We think that this will also prove to be useful e.g. in multi-domain environments.

Components and interfaces of the Management System are depicted in figure 2.

Path computations performed by the Management System depend on accurate information of the network topology. This infromation is collected from topology information distributed by the Control Elements by listening in to the topology messaging. The topology information is distributed with IS-IS routeing protocol, modified with Ethernet extensions [15]. This information is collected in the MS's database, where it can be accessed by the path computation element, CE manager, and any external management plane applications by using the SOAP function calls supported by the MS.

Path computation itself is based on a naive algorithm, and can currently support both point-to-point paths with optional protection schemes (1:1, 1+1) selected, and point-to-multipoint trees with non-disjoint 1+1 protection. More robust path calculation algorithms is one of the areas where further development would be warranted [16].

## *B. Control Element*

Control Element (CE) controls a single forwarding capable node in the network. Each CE is paired with the corresponding Forwarding Element (FE), and all the CEs are managed by a single MS. Signalling between CEs is handled by RSVP-TEEth messaging. CE controls the corresponding FE with IETF ForCES protocol [17]. The communication between CEs and the MS is handled with mRSVP.

A single Control Element includes a local forwarding database, i.e. information of all the tunnels traversing the node, and their next-hop destinations. In addition to this, the CE also maintains tunnel state necessary for signalling, and keeps track of the network health in the immediate vicinity, as well as the health of all the tunnels originating or destined to the node. The different elements of CE and their interactions are illustrated in figure 3.

Different CEs communicate with each other by using extended RSVP messages (RSVP-TEEth). CE-to-CE communications are limited to path reservation and teardown – all other

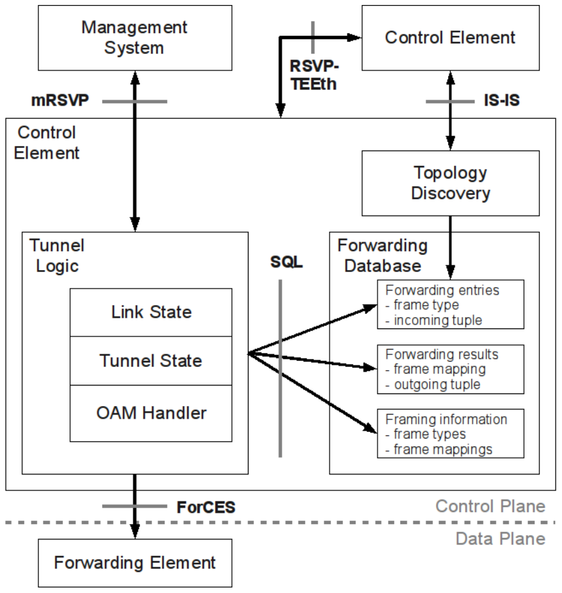


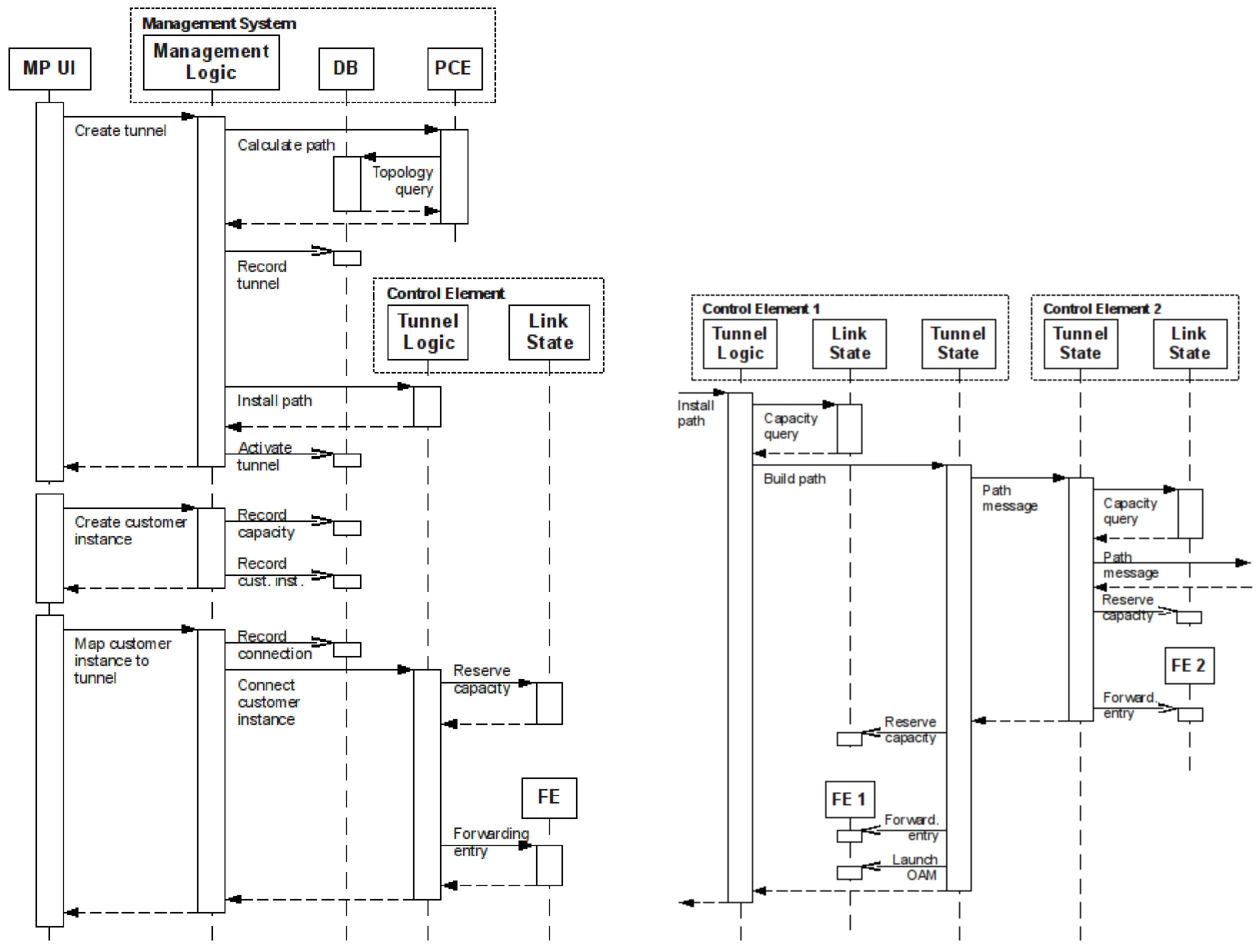
Fig. 3. Control Element components

network operations are handled either by CEs independently, via Management System, or with lower layer OAM functionalities described below. Path reservation is always initiated by the MS and reservation is done from the source CE towards the destination. Path teardown is also initiated by the MS, but the messaging can originate from any CE along the path, as dictated by the MS.

The CE communicates with an FE using the ForCES protocol. Our prototype is currently limited to one-to-one relationships between CEs and FEs. The FEs we are using are EZChip based network processors running software written by BGU for this purpose [13]. The FEs are customizable in terms of supported Ethernet frame types and forwarding logics, which allows us to use custom frame types and two-stage forwarding lookups required by the architecture.

Monitoring of the network and tunnel health is done with standard OAM mechanisms as defined by ITU-T [18]. We are currently implementing only link and tunnel level OAM. The OAM processing is mainly offloaded to the FE, as the OAM traffic needs to be generated with millisecond precision. The CE receives notifications of failed OAM from the FE and acts on them according to installed policies, e.g. by switching traffic to a protection path.

Another important task of the Control Element is to discover the network topology and spread the topology information around in the network. For this, we are using a modified Quagga IS-IS daemon [19]. The IS-IS daemon uses information available from FE: FE capabilities, interfaces and their properties, etc. and spreads this in the network inside IS-IS LSPs. This information is also used for bootstrapping the network, i.e. calculating the routes for management VLAN connecting all the CEs and the MS.



(a) Creating a tunnel, connecting customers

(b) Installing a tunnel in the network

Fig. 4. Flow of control during tunnel creation

## III. Basic Operation

We now describe the basic operation of creating a tunnel in the network and adding a customer to it. The premise is that the network has had time to stabilize after bootstrapping, and the topology has been updated to Management System database (DB). Some tunnels might already exist in the system, but no suitable tunnel for this connection is available. Customers locations are known, but their connection parameters to the network have not been defined. Available service classes have also been defined by the network operator. The process is controlled from an UI on the management plane. The basic flow of control is depicted in figure 4a.

First, the UI issues a tunnel creation request over SOAP interface. This request is handled by management logic (ML) on the Management System. The request contains information of the tunnel's endpoints, capacity and service class. The ML then asks the path computation element (PCE) to calculate a suitable path for the tunnel.

PCE queries DB about the topology and calculates a suitable path, along with any requested protection paths, for the tunnel. The PCE then returns this information to the ML which records the path to the DB. The ML then issues an install path command with mRSVP to the ingress Control Element on the path (CE-1), with full information of the path.

CE-1's tunnel logic processes the path installation command as depicted in figure 4b. After the path installation is complete CE-1 messages the MS's ML which can now activate the tunnel in the DB and inform the UI of succesful operation.

When CE-1 receives path installation instructions via mRSVP they are first processed by tunnel logic (TL). TL queries link state (LS) for available capacity. If there is enough capacity, the TL then instructs tunnel state (TS) to begin building the path.

TS looks at the path information, records it, and constructs an RSVP-TEEth Path message to be sent to the TS of the second Control Element on the path (CE-2). The TS of CE-2 checks its local LS for available capacity, records the message, modifies the Path message with its own information, and sends it forward along the path. If there is an error, e.g. not enough free capacity, CE-2 responses with an error message.

When the Path message arrives to the egress CE of the tunnel, the processing differs slightly. First the TS of the egress CE does the necessary capacity checks and reservations. Next the TS installs a forwarding entry to the local database, and to the connected FE via ForCES, instructing the FE of any OAM used on the tunnel, if applicable. Finally the TS sends back a Resv message. This message is sent along a reversed path with relation to the Path message. When a Resv message arrives to a CE, it triggers the CE's TS to reserve capacity for the tunnel from the local LS. The TS also records a forwarding entry to the local database, and instructs the connected FE via ForCES with the forwarding entry. After this, the Resv message is sent onwards.

Once the Resv message reaches the ingress CE's TS, the TS reserves capacity, installs a forwarding entry, and launches OAM if requested. Then the TS returns TL the succesfully built path, and the TL can in turn inform the MS's ML of complete path installation.

After the tunnel has been installed we need to connect customers to it. This is a two-phase process, where the UI needs to first tell the system how exactly the customer is connected and then attach the connection to an existing tunnel. This happens by creating a customer instance (CI). CI defines the interface the customer is connected on, and the frame type and possible tags the customer is sending towards the operator, e.g. 802.1Q with VLAN number 128. The CI also includes limits on the bandwidth the customer is allowed to use when connecting through it. It is worth noting, that there can be more than one CI per customer per interface. When the UI issues customer instance creation command via SOAP, the ML of the MS records both capacity reservation on the connecting interface and other CI parameters in the DB.

After a CI has been created on an end point node of the tunnel, the UI needs to issue a SOAP command that maps the CI to the tunnel. ML handles this command by first recording the connection to MS's DB and then issuing an mRSVP instruction to the end point CE to connect the customer to the tunnel. The CE does not store information about the CI anywhere, but instead uses the information of customers frame type and tags, connection interface, and existing tunnel to create a forwarding entry for the FE. The CE's TL first reserves the capacity from the interface for the connection from the LS. Then the TL stores a forwarding entry to CE's local database, and sends the forwarding entry to FE via ForCES. After the entry has been installed to the FE, the TL informs ML, which in turn can notify UI of a succesful operation. Naturally a CI needs to be created and connected on both ends of the tunnel for any traffic to pass through.

## IV. Conclusions and Future Work

Our implementation shows that it is possible to create a simple and yet flexible control plane for a service oriented transport network on top of Ethernet. By modifying the existing Ethernet frame structures we can easily support a network architecture, which allows for manageable and efficient transport, and at the same time hides the transport network mechanics from the end user. By using existing, established technologies, and extending industry standard protocols, we can offer a versatile platform for future network services.

Our next goal is to mature the control plane software to a point, where it, together with the data plane developed by BGU, can be used to offer a stable platform for creating more advanced network services. When the control plane proves robust enough, we can extend further into the transport service and network service layers of the ETNA network architecture. For example, the system already supports a coarse level of mobility in the core, which follows from the transport layer P2MP primitives. By adding a network service layer support for mobility and user identities, we believe that this technology can be used to deliver both network and end user level mobility from layer 2 upwards.

Our implementation and architecture also compares with MPLS-TP [20]. It already supports most of the requirements MPLS-TP sets on control plane operation as is. By changing the Forwarding Elements to ones compatible with both MPLS-TP and ForCES, we see no obstacles to using our control plane with MPLS-compatible data plane.

## References

[1] *Internet World Stats*, http://www.internetworldstats.com/stats.htm, checked July 22, 2009.

[2] *GSM World - Market Data Summary*, http://www.gsmworld.com/newsroom/market-data/market_data_summary.htm, checked July 22, 2009.

[3] *World Broadband Statistics Q1 2009*, http://point-topic.com/content/operatorSource/dslreports/World+Broadband+Statistics+Q1+2009.pdf.

[4] *The Future of the Internet Economy - A Statistical Profile*, OECD Ministerial Meeting on the Future of the Internet Economy, Seoul, South Korea, 2008. http://www.oecd.org/dataoecd/44/56/40827598.pdf

[5] W. Yue, *The role of emerging broadband technologies on the converged packet-based network*, Optical Fiber Communication Conference, 2006.

[6] *IPv4 Address Report*, http://www.potaroo.net/tools/ipv4/index.html, checked July 22, 2009.

[7] A. Feldmann, *Internet clean-slate design: what and why?*, ACM SIGCOMM Computer Communication Review, Vol 37, issue 3, 2007.

[8] D. Allan, *Ethernet as Carrier Transport Infrastructure*, Communications Magazine, IEEE, Vol 44, issue 2, 2006.

[9] *ETNA*, http://www.ict-etna.eu/index.html.

[10] R. Kantola, M. Luoma and O.-P. Lamminen, *Transport for Carrier Grade Internet*, submitted for BIPN'09

[11] J. Ryynänen, *Routed End-to-End Ethernet Network – Proof of Concept*, Master's Thesis, Helsinki University of Technology, 2008.

[12] *ETNA - Network and service architecture*, http://www.ict-etna.eu/documents.html.

[13] *ETNA - Simulation and prototyping goals, requirements, and models*, http://www.ict-etna.eu/documents.html.

[14] A. Csaszar et al., *Converging the Evolution of Router Architectures and IP Networks*, Network, IEEE, Vol 21, issue 4, 2007.

[15] T. Toropainen, *A Routing Protocol for Ethernet Transport*, Master's Thesis, Helsinki University of Technology, 2008.

[16] V. Holopainen and R. Kantola, *Tackling the Delay-Cost and Time-Cost Trade-Offs in Computation of Node-Protected Multicast Tree Pairs*, in Proceedings of APNOMS'09, 2009.

[17] L. Yang et al., *Forwarding and Control Element Separation (ForCES) Framework*, IETF Network Working Group, RFC 3746, 2004

[18] *OAM functions and mechanisms for Ethernet based networks*, ITU-T, Y.1731, 2006.

[19] O. Santolalla, *Implementation of IS-IS Extensions for Routed End-to-End Ethernet*, Master's Thesis, Helsinki University of Technology, 2009.

[20] B. Niven-Jenkins et al., *MPLS-TP Requirements*, IETF MPLS Working Group, Internet-Draft, 2009