Experimental Validation and Assessment of Multi-domain and Multi-layer Path Computation

Sebastian Gunreben[1], Ramon Casellas[2], Ricardo Martinez[2], Raul Muñoz[2], and Joachim Scharf[1]

[1] Universität Stuttgart - IKR, Pfaffenwaldring 47, 70569 Stuttgart, Germany
gunreben@ikr.uni-stuttgart.de
[2] Optical Networking Area, Centre Tecnològic de Telecomunicacions de Catalunya, CTTC, Av. Canal Olímpic s/n, 08860 Castelldefels (Barcelona), Spain

Abstract. Within the framework of the BONE European Network of Excellence, we setup a multi-domain multi-layer testbed covering three different networks at two distinct locations in Europe. The testbed includes two Ethernet switched client networks, which are interconnected by a wavelength switched server network. Each of these networks is operated by a GMPLS control plane and implements a path computation entity, following either the IETF PCE proposal or the DRAGON NARB. Since the communication protocols of IETF PCE and DRAGON NARB are incompatible, we propose and develop an application layer gateway, enabling inter-domain path calculation.
In this paper, our contributions are three-fold: First, we provide a comparison of both communication protocols. Second, we present the architecture and working principles of the designed NARB/PCE Gateway, specifying the available features and constraints of our implementation. Third, we validate, for the first time, the PCE/NARB connectivity while evaluating the performance of a path computation request in terms of request response time in the multi-domain and multi-layer testbed.

Keywords: multi-domain, multi-layer, PCE, path computation.

1 Introduction

The ITU-T defined the term next generation network (NGN) in [9]. One of their major objectives is the separation of the transport network (transport stratum) and the service platform (service stratum). The transport stratum enables a variety of transport network technologies, e. g., connection (wavelength) or packet switched networks (Ethernet [7], Internet Protocol). Networks operating and controlling multiple technologies in parallel refer to multi-layer networks. Therein, the underlying transport technology is in general abstracted by a virtual topology. Any layer operating on top of this virtual topology may request additional connectivity from the transport technology. In general, these transport networks with different underlying connection-oriented technologies

T. Magedanz et al. (Eds.): Tridentcom 2010, LNICST 46, pp. 347–362, 2010.

required separate controlling entities. The GMPLS control plane framework [13] is one proposal for a unified integrated control plane, i. e., operating various technologies on different layers. It is responsible for the establishment, management and release of end-to-end connections, such as optical connections or lightpaths, commonly referred to as lambda switched capable Label Switched Paths (LSP). Such a control plane includes the functionalities of: a) routing and topology information dissemination, b) path computation and c) LSP signalling. Including several layers within one single control plane instance increases the number of traffic engineering links (TE-links) within one routing domain. Any path computation, which operates on these TE-links, becomes complex and requires powerful processing engines.

Additionally, as the ITU-T proclaims the NGN as the implementation of a global information infrastructure [9, 8], it demands inter-operable networks throughout different network operators and network technologies. Consequently, paths traversing multiple domains refer to a multi-domain scenario, and any path computation involves the controlling entities within each of these domains. These computation tasks may add additional constraints to the path computation, which further increase the complexity.

Summarizing, the path computation task becomes complex and extensive in multi-domain and multi-layer networks. Consequently, the IETF proposed the Path Computation Element (PCE, [5]), i. e., an explicit entity to perform path computation within these transport networks. They defined the requirements, the architecture [5] and a communication protocol [14] for the path computation function. In parallel to the IETF, the US DRAGON project [11] also proposed a Network Aware Resource Broker (NARB, [15]), which serves the same purpose, but provides a different communication protocol.

In this paper, we present the experiences of interoperating PCE and NARB devices within a multi-domain, multi-layer network infrastructure. For the first time, we present a successful path computation covering three different domains (located in Spain and Germany) and two different transport layers (an Ethernet switched network and a Wavelength Switched Optical Network, WSON). Our contribution is three-fold. For the interoperation of PCE and NARB, we first provide a comparison of both communication protocols; second, we present the architecture and working principles of the designed NARB/PCE Gateway, specifying the available features and constraints of our implementation. Third, we evaluate the performance of a path computation request in terms of request response time in the multi-domain and multi-layer testbed.

The next section introduces and classifies the related work in the field of multi-layer, multi-domain path computaion. Section 3 introduces the implementations of the two path computation elements used in our scenario. Section 4 presents the architecture and functionality of our PCE/NARB Gateway and compares the PCE and the corresponding NARB protocol. Section 5 introduces our testbed with three different domains on two locations in Europe. We evaluate the performance of the whole system in the result section 6. Section 7 summarizes our contribution.

2 Related Work

We classify the related work in multi-layer and multi-domain studies and focus on the applied methods. Multi-layer path computation strategies were the subject of numerical simulation studies of Cugini et al. in [4] and Gunreben/Rambach in [6]. Multi-domain path computation was subject of the prototypical implementations of Bianzino et al. [2] and Casellas et al. in [3]. This paper provides both, prototypical implementation in a multi-layer and multi-domain testbed.

3 Path Computation Elements

This section introduces the architecture and functional details of both path computation elements applied in our testbed. Firstly, we describe the IETF PCE implementation of CTTC. Next, we present the proprietary PCE solution from the DRAGON project. Thereby, both sub-sections include a general introduction, a description of the architecture and a brief summary of the application program interface (API). Finally, we compare that part of the APIs, which is relevant for our scenario.

3.1 IETF PCE Implementation

The IETF proposes in [5] and [14] the architecture and the API/communication protocol for path computation elements. This section gives an introduction and a brief summary of both. A working implementation of a PCE in CTTC's ADRENALINE® testbed shows the applicability of this IETF solution.

Introduction. In general, the IETF PCE proposal covers generic PCE-based implementation building blocks, such as composite, external, and multiple PCE path computation approaches. The retained architecture in this work involves an external PCE, available for multiple PCE path computation tasks.

Further, the aforementioned normative documents discuss architectural considerations including centralized and distributed computation, synchronization, PCE discovery and load balancing. The PCE provides the following additional functions: detection of PCE aliveness; communication between Path Computation Clients (PCC) and the PCE; PCE-PCE communication; Traffic Engineering Database (TED) synchronization; monitoring; policy and confidentiality enforcement.

Architecture. CTTC implemented the PCE according to the IETF standard. Fig. 1 shows the architecture of the implementation. It consists of three major blocks, (a) the management and processing of Path Computation Element Communication Protocol (PCEP) messages from PCC and PCE peers, (b) the management and update processing of the traffic-engineering database (TED) and (c) the path calculation itself. Multiple asynchronous processes (threads) realize the functionality of these blocks.

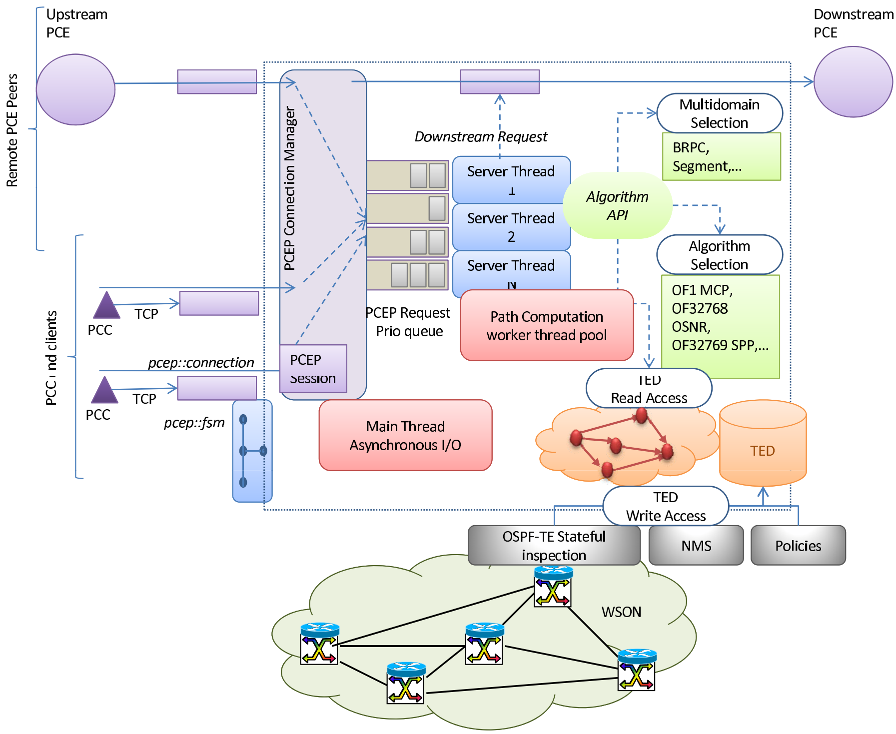

Fig. 1. Functional Architecture for CTTC PCE

The PCE main thread is responsible for managing PCEP connections. It queues incoming path computation requests in a priority-based queue. A pool of threads (with a configurable number of worker threads) serves these requests from the queue. In case of a multi-domain scenario, the PCE requires, besides the local topology, a downstream response from a peer PCE. This dependence blocks this thread until it receives the computation answer from the downstream PCE, controlled by a timeout mechanism. Thereby, a timeout mechanism avoids infinite blocking. In case of timeout, the PCE replies with a NO PATH object.

The PCE implementation provides an interface to allow different algorithms for path computation as plug-ins. This leverages granular, per-request algorithm selection (using Objective Function - OF - codes to identify specific algorithms), based on client preference and pre-defined policies. The common interface of these path computation algorithms involves: a) access to the abstracted TED in form of a directed graph; b) the possibility of requesting path computations from an external (e. g., downstream) PCE and c) access to dynamic / static information covering network reachability and pre-configured PCE domain chains.

In our scenario, the TED includes the optical network topology including link and node GMPLS TE attributes along with specific optical networks extensions such as the number of available optical signal regenerators and optical signal to noise ratio (OSNR) figures. One or more dedicated threads update this TED. Thereby, the TED synchronization mechanism is non-intrusive. The PCE monitors the OSPF-TE [10] traffic and constructs the database by means of stateful inspection of the OSPF-TE link state updates. This approach passively reuses the OSPF-TE dissemination mechanism and does not require the creation of an additional listener adjacency.

PCEP API. The main interface to access path computation services from a PCE is the PCEP protocol [14]. PCEP allows PCCs to request path computations by means of a message oriented protocol over a TCP connection [12]. The PCEP protocol defines seven basic messages, which we classify in three different categories: session management, path computation and exception handling. The evaluation in section 3.3 studies the detailed messages and compares them to the NARB equivalents.

A special feature of PCEP is the session management. After an initial session setup, the lifetime of this session follows one of two different modes. In the *persistent mode*, the lifetime can span several requests. In the *non-persistent mode* the session ends after a single request. Additionally, the path request message (PCReq) may include one or several requests, allowing for synchronized and dependent path computation.

3.2 DRAGON NARB

Besides the IETF initiative, the US project DRAGON (Dynamic Resource Allocation via GMPLS Optical Networks, [11]) proposed and implemented a device with similar functionality than that of a PCE. The project designed and implemented a Network Aware Resource Broker (NARB, [15]), which performs path computation in multi-layer and multi-domain networks. Comparable to PCEP, NARB also provides an API for remote path computation requests. The next two sections introduce the NARB architecture as well as the API protocol.

Introduction. The NARB represents a path computation element within an Autonomous System (AS). It consists of two entities, the NARB itself and the Resource Computation Element (RCE). NARB provides higher-level functions like topology abstraction for an inter-domain scenario or inter-domain path computation. In an inter-domain scenario, the NARB elements of each domain interconnect and exchange static reachability information. For security and competitive reasons, the exchanged topology information may only provide a subset or abstracted view of the real topology. A RCE enables path computation and provides a raw database of the topology.

Architecture. The NARB provides inter-domain as well as intra-domain path computation functionality. It obtains the intra-domain topology information from listening to the local OSPF-TE routing instance and the inter-domain topology information from listening to the inter-domain OSPF instance. Besides, the NARB software provides the following interfaces:

– Internal interface to RCE for path calculation
– API interface for path computation requests
– External interface to peer NARB of other domains

For a detailed view on the NARB architecture, we refer the reader to the documents published by the DRAGON consortium [11, 15]. The next section introduces the application programmable interface for inter-active usage of the NARB functionality.

NARB API. We refer to [15] for a complete feature list of the NARB API. This section covers the major functions exploited within this study. The NARB API clients connect to the NARB API server on a dedicated port. Thereby, it allows several simultaneous connections. The NARB API message structure shows a header and a message body. The header includes, among others, the message type and options. The body includes type-length-value (TLV) encoded data.

The message type distinguishes seven different messages, which belong to two different categories. The first class includes messages related to path request and reply. Resource reservation related messages make up the second class. While [15] provides details on all these messages, we do not use all types of messages in our scenario. Especially the reservation messages are out of scope of this paper. The following four message types are relevant for our study.

– *Client LSP Query Request* indicates a path computation request from a client.
– *Peer LSP Query Request* indicates a recursive path computation request from NARB to its peer NARB in a multi-domain environment.
– *LSP Query Reply* with an Explicit Route Object (ERO) indicates a successful path computation task, and the reply includes the explicit route object required for the signalling process.
– *LSP Query Reply* with ERROR indicates an unsuccessful path computation task. The message includes the reason for this error, e. g., no source, destination, no route or internal errors.

3.3 Comparison of PCEP and NARB API

In this section, we compare NARB API and PCEP. However, our comparison is neither exhaustive nor complete as we restrict ourselves to the basic functionality for path computation in a multi-domain and multi-layer scenario. Table 1 compares both protocols with respect to the path computation scenario. The table classifies the protocol features in three different classes: session management, path computation and exceptions. The left column gives the functionality. The second and the third column depict the realization of this functionality in PCEP and NARB API, respectively.

Both, PCEP as well as the NARB API rely on TCP. On top of TCP, PCEP implements a session management including Open, Close and KeepAlive messages. The Open messages enable the negotiation on different parameters (deadtime, keepalive timer) during the session initiation phase. The KeepAlive messages provide mechanisms to check if the PCE is still alive and operable. Besides

Table 1. Comparison of PCEP and NARB API

Functionality	PCEP	NARB API
Session management		
Open Message	Session negotiation	-
KeepAlive Message	Session liveness monitoring	-
Close Message	Session termination	-
Identifier	Session identifier	Universal client id, sequence number
Path computation		
Computation request	May include several requests per message	One request per message
Constraint request	support of end-points constraint	support of end-points constraint
	support of bandwidth constraint	support of bandwidth constraint
	-	Requires technology constraints (encoding, switching type)
	Can exclude objects (XRO)	-
Computation reply	successful reply includes ERO	successful reply includes ERO
	NO PATH object to indicate no path	Error message indicating no path found
	May include several replies per message	One reply per message
	Multiple alternative paths per reply	One reply per message
Exceptions		
Error messages	several error classes	one error class
Notification	exceptional signalling for unforeseen events	-

this, they acknowledge the Open message during session establishment. The Close message terminates a session. The NARB API does not provide any session management at all. After the TCP session is established, the client may pass the NARB API message directly to the NARB server. With respect to the response time of the initial request, the reduced session overhead leads to a reduced response time.

Path requests also provide similarities as well as differences. PCEP allows adding multiple path requests within a single path request message. In contrast to this, the NARB API expects only one request per message. Both protocols allow specifying constraints for path computation. While PCEP allows indicating optional or mandatory constraints, the NARB API requires certain parameters in any case, i. e., encoding type, switching type, and bandwidth. The only exception in the NARB API is the quality of the hops. The client may specify if hops may be strict or loose and if this constraint is mandatory.

The path computation reply messages are very similar. In case of successful path computation, the reply messages of both protocols contain standard compliant explicit route objects (ERO, [1]). The difference is again the number of

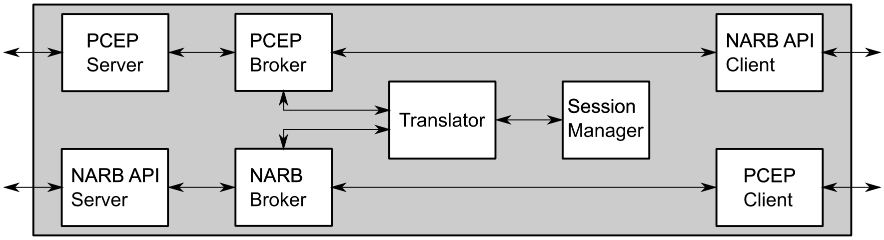

Fig. 2. NARB/PCE Gateway architecture

possible paths per reply message. While PCEP allows multiple responses, each containing one or more paths and attributes, the NARB API only allows one.

In addition, the error handling is different in case that no path is found. While PCEP replies with a NO PATH object in a message with optional TLVs, the NARB API message only indicates the reason for the unavailability of a path. Both protocols implement the errors unknown path source/destination and no path could be computed.

At any time an error may occur. PCEP as well as the NARB API provide mechanisms to indicate such errors. The classification of the NARB API errors includes mainly two error classes: unavailable paths and NARB internal errors. Additionally, PCEP includes classes for policy violations, malformed messages and non-conforming PCEP requests. Summarizing, the error classes of both protocols are incompatible, the commonality may be a most generic error.

4 PCE/NARB Gateway

The previous section highlighted the major differences of both protocols. IETF PCEP and the DRAGON NARB API are in general incompatible as they show different message formats and protocol states. Nevertheless, on a functional level they are compatible. Any interoperation requires an application layer gateway, which implements both interfaces and performs a translation of messages. It implements at the interface to the NARB the NARB API while on the interface to the PCE it implements the corresponding PCEP interface. On both sides, server instances of both protocols reside. As a path computation request from a NARB translates to a path computation request to the PCE, the gateway also implements the client functionality for NARB API and PCEP.

We implemented the NARB/PCE Gateway using the Java programming language for platform independence. The next three sections introduce the architecture and working principle of the gateway and specify the available features and constraints of our implementation.

4.1 Architecture

Fig. 2 depicts the gateway architecture with the according building blocks. The following paragraphs introduce the functionality of each block.

The PCEP and NARB API server are responsible for the TCP connection management on both sides, i. e., they accept connections and create sockets. On a successful connection, they forward the socket information to the corresponding broker instance.

The broker represents the instance, which associates a NARB connection with a PCE session. It reads messages from the socket and interprets these messages. It finally forwards these messages to the translator block.

The translator block is responsible to convert a PCEP message to an equivalent NARB API message and vice versa. Thereby, it translates the common feature-set of a message with respect to the other protocol. The translator is able to translate messages required for a path computation request including session and error handling. Besides, the translator employs the functionality of the session manager.

NARB API and PCEP apply different identifiers for a session (cf. Tab. 1). However, an entity needs to associate replies from one protocol to the requests of the other protocol and vice versa. The session manager implements this functionality. It maintains a list of identifiers for both, NARB API and PCEP requests. Besides, the list contains the information on the corresponding broker instance.

The client instances of each protocol forward the translated messages to the peer NARB/PCE, after necessary connection setup. The next section introduces the working principle in detail.

4.2 Working Principle

Fig. 3 illustrates the working principle of the gateway. The starting point is a path computation request using PCEP. Therefore, the PCEP client connects to the PCEP server instance at the gateway and establishes a TCP connection. After successful connection establishment, the PCEP server passes the information on the socket to the PCEP broker. The PCEP client and the PCEP broker perform the handshake mechanism using the PCEP open sequence (two times Open and KeepAlive, [14]).

After a successful handshake, the PCEP client sends a request to the gateway. The PCEP broker receives the request and translates the request in an internal data structure. It passes the internal data structure to the translator entity. The translator maps the internal data representation into an equivalent representation of a NARB API message. Thereby, it applies the session manager to record the identifiers used in both protocols and PCEP broker instances. The PCEP broker receives back the translated message from the translator. It forwards the NARB request to the NARB client instance, which connects to the peer NARB server and forwards the request to it.

After processing the request on NARB side, the NARB client may receive a response message. It parses the response message and checks with the Session

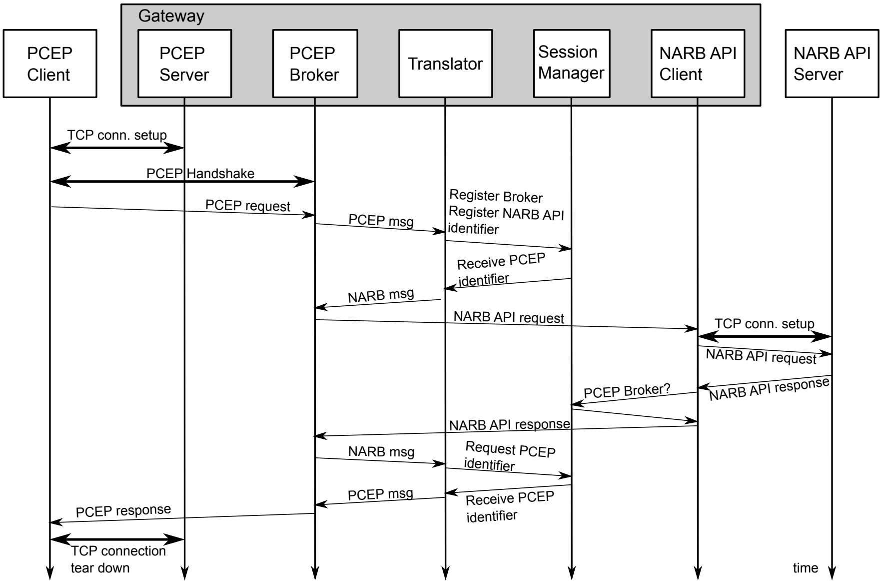

Fig. 3. NARB/PCE Gateway working principle

Manager for the corresponding PCEP broker. With this information, the NARB client forwards the message to the PCEP broker. The latter one uses again the translator to receive a PCEP response message. The Translator performs the translation from NARB API response to PCEP response and requests the session manager for the corresponding sequence number on PCEP side. The PCEP broker passes the complete NARB API message to the requesting PCE client.

The same procedure applies for NARB API requests, which the gateway translates to PCEP messages. The message sequence chart is the same, except the open sequence, which occurs after translating, when forwarding to the peer PCE server.

4.3 Evaluation

This section presents a brief evaluation of the gateway implementation. On a system perspective, the gateway is a multi-threaded program. It operates one main thread for network input/output and maintains for each peer an additional thread. Within this thread, the gateway receives and sends messages and performs the translation tasks as described before. This thread is non-blocking, i. e., the thread does not require to wait for an answer from a downstream peer for further processing. Incoming messages are processed in a first-come first-service discipline per peer NARB/PCE.

This single thread per peer NARB/PCE remains open until a dedicated request tears down this session, i. e., the gateway implements the persistent mode

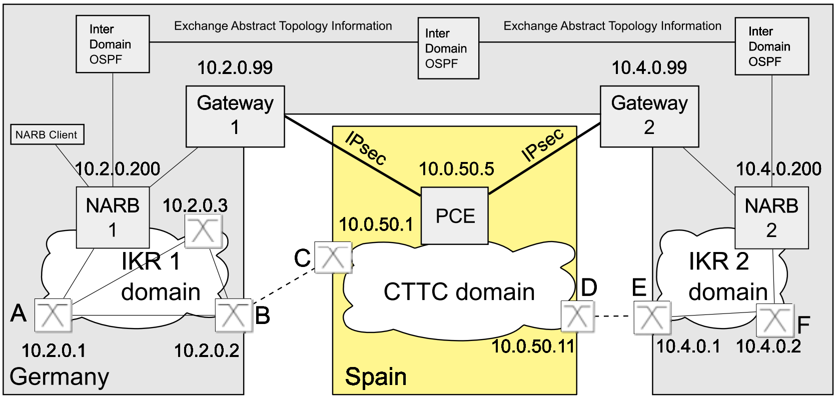

Fig. 4. Multi-domain, multi-layer testbed

of PCEP. For simplicity, the gateway announces a value of 0 for both keepalive and deadtimer, i. e., it keeps the connection open until arrival of a close request.

For multi-domain path computation, the PCE requires a METRIC object [14] in the path computation reply message. This object includes the value of a metric classifying the replied path, e. g., length in number of hops. As the NARB does not support such METRIC Object, the gateway adds this object to the reply from the NARB and forwards it to the PCE.

5 Multi-layer Multi-domain Testbed

This section presents our multi-layer and multi-domain testbed to evaluate both, the inter-PCE connectivity using the PCE/NARB gateway and the performance of a path computation request. The first section introduces the testbed setup while the second section evaluates a sample for a path computation request.

5.1 Introduction

The testbed setup (cf. Fig. 4) interconnects three domains, the IKR 1 and IKR 2 and the CTTC domain. The testbeds of IKR 1 and IKR 2 reside at the University of Stuttgart, Germany. Both testbeds implement an Ethernet based data plane and the DRAGON/GMPLS based control plane [11]. Thereby, the data plane is connection-orientated using Ethernet VLAN tagging [7]. Each domain represents an individual OSPF-TE area and realizes the path computation task with a NARB element in each network.

The ADRENALINE® testbed resides at the premises of CTTC in Castelldefels (Barcelona) in Spain and implements a Wavelength Switched Optical Network (WSON). The optical layer is GMPLS-controlled and shows 14 network nodes with an emulated optical hardware. The path computation function has

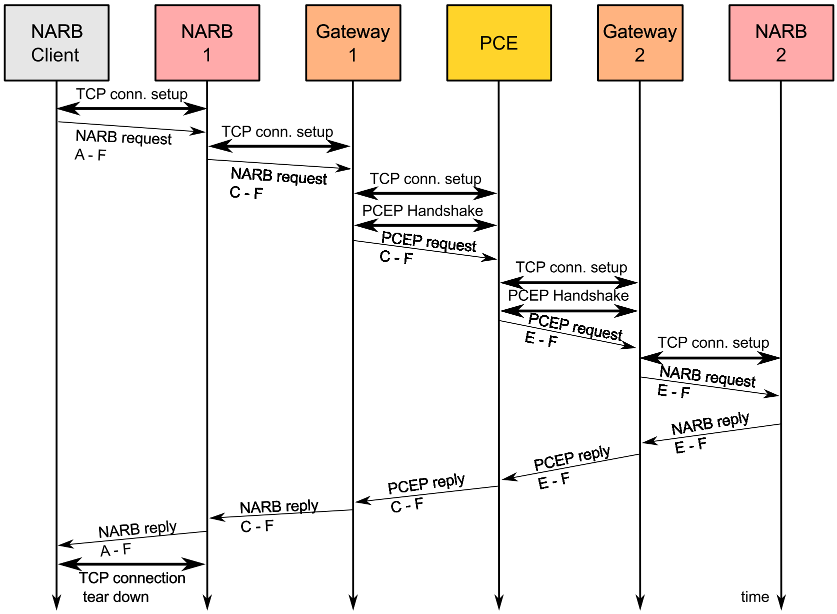

Fig. 5. Path computation request in our inter-domain testbed

been centralized in a single Path Computation Element within the domain, which spans a single OSPF-TE area.

Both, NARB and PCE receive intra-domain topology information through passive listening to the OSPF-TE [10] protocol. Between these networks, they exchange inter-domain topology information, which corresponds to the abstract/summarized topology as introduced in section 3.2. The abstract topology information includes only the border nodes of each network hiding the nodes and TE attributes within the network. For simplicity, we emulate the OSPF adjacency to the ADRENALINE® instance to the CTTC domain by a third OSPF instance at the IKR testbed. This additional OSPF adjacency announces the abstract topology of the CTTC domain deployed using the ADRENALINE® testbed (nodes C and D) to the OSPF instances of networks IKR 1 and 2.

As both show non-compatible protocols, a PCE/NARB gateway resides between both (Gateway 1 and 2). An IPsec tunnel via the public Internet realizes the interconnection between the NARB/PCE gateway and the PCE of the ADRENALINE® testbed. For NARB 1 and NARB 2, the gateway acts like a peer NARB. For the PCE, the gateways act like peer PCEs.

5.2 Multi-domain Multi-layer Request

This section presents a sample path computation request for inter-domain path computation. Fig. 5 depicts the message sequence chart of a typical request covering all three domains and the intermediate devices. The message sequence chart assumes unestablished TCP and PCEP connections between all entities. The request asks for a suitable path from node A to node F of Fig. 4. The request does not include any other constraints than the endpoints.

The NARB client from the DRAGON suite performs the path computation request. After the initial TCP handshake, the NARB client sends the request to NARB 1 using the NARB API. NARB 1 receives the request and splits it in two parts. While NARB 1 handles itself the local part, i. e., the path computation from node A to node B, it forwards the remote part to the downstream NARB, i. e., Gateway 1. Therefore, NARB 1 establishes a TCP connection first. The gateway performs a translation of the NARB message, which results in a PCEP request for the downstream PCE. Besides, the gateway records the identifiers of the received NARB message and the PCEP message to later identify the response to the correct request. Before forwarding the request to the remote PCE, the gateway performs the TCP and the PCEP handshake. The PCE receives the request and splits the request in a local part (path from C to D) and a remote part. After the TCP and PCEP handshake, it forwards the remote part to the peer PCE, i. e., Gateway 2. Again the gateway translates the request in a NARB conform way, records the identifiers and forwards the request after the TCP handshake to NARB 2. Now, the request of NARB 2 only includes local nodes E and F.

After computation, NARB 2 responds with a suitable path to the peer NARB, i. e., Gateway 2. After passing the gateway, the PCE receives the response and joins the results from the local path computation and the results from the downstream PCE, i. e., NARB 2, and forwards the result to the requesting entity, i. e., Gateway 1. After passing the gateway, NARB 1 receives the responses of PCE and NARB 2, joins them and forwards the outcome to the requesting NARB client. After reception of the response, the NARB client terminates the TCP connection, as there is no further request. The other TCP and PCEP connections remain open, as there is no indication to close them.

If there is any error on this path, the response in upstream direction includes an error message, which is translated in an appropriate format. As the error reasons do not overlap, in most cases a general error of no path found occurs in the response.

6 Performance Evaluation

For a quantitative performance analysis, we evaluated the time from sending a request until receiving the response. For this evaluation, we analyzed two different scenarios, the PCEP non-persistent and persistent mode.

In PCEP non-persistent node, for every request, the PCE establishes a new TCP and PCEP connection to NARB 2. The PCE closes this PCEP connection after receiving the response from NARB 2. For a new request, the whole setup procedure is performed anew. The connection from NARB 1 to the PCE remains persistent.

Nc	Time	Source	Destination	Protocol	Info
1	0.000000	10.0.50.5	10.2.0.99	PCEP	OPEN MESSAGE
2	0.000280	10.2.0.99	10.0.50.5	PCEP	OPEN MESSAGE
3	0.053381	10.0.50.5	10.2.0.99	PCEP	KEEPALIVE MESSAGE
4	0.053782	10.2.0.99	10.0.50.5	PCEP	KEEPALIVE MESSAGE
5	0.116968	10.2.0.99	10.0.50.5	PCEP	PATH COMPUTATION REQUEST MESSAGE
6	0.171188	10.0.50.5	10.4.0.99	PCEP	OPEN MESSAGE
7	0.225655	10.4.0.99	10.0.50.5	PCEP	OPEN MESSAGE
8	0.225715	10.0.50.5	10.4.0.99	PCEP	KEEPALIVE MESSAGE
9	0.279501	10.4.0.99	10.0.50.5	PCEP	KEEPALIVE MESSAGE
10	0.279558	10.0.50.5	10.4.0.99	PCEP	PATH COMPUTATION REQUEST MESSAGE
11	0.337972	10.4.0.99	10.0.50.5	PCEP	PATH COMPUTATION REPLY MESSAGE
12	0.338104	10.0.50.5	10.4.0.99	PCEP	CLOSE MESSAGE
13	0.338509	10.0.50.5	10.2.0.99	PCEP	PATH COMPUTATION REPLY MESSAGE

Fig. 6. Trace of PCEP setup and response/request messages at PCE

In PCEP persistent mode, the PCE also keeps the PCEP connection to NARB 2 open and reuses this connection for further requests. Only the first request performs the setup procedure, while the PCE forwards subsequent requests immediately. The mechanism to keep the PCEP connection open corresponds to the KeepAlive messages and deadtimer agreed in the open sequence.

6.1 Non-persistent Mode

In non-persistent mode, the PCE requires a TCP/PCEP setup between PCE and NARB 2 for each request (cf. Fig. 5). Thereby, the round trip time (RTT) between the two instances is the main driver for duration of the setup. In our scenario, we measure a RTT of about 52 ms between the IPsec router at IKR and the IPsec router at CTTC. We also observe some jitter as the IPsec tunnels pass several thousands of kilometres through the Internet.

Fig. 6 illustrates the PCEP messages at the PCE including the setup and request/response messages. The current implementation of the PCE (IP: 10.0.50.5) sends as direct consequence of the TCP connection establishment a PCEP Open message to Gateway 1 (IP: 10.2.0.99) (line 1). Gateway 1 also sends a PCEP Open message after successful TCP connection establishment. Due to the three-way handshake of TCP and the different times of connection establishment, the PCE receives this message from Gateway 1 more or less immediately after sending its own OPEN (line 2). At the beginning, the TCP window size allows only one outstanding unacknowledged packet. Therefore, both entities need to wait on the TCP acknowledgment before being able to send a further PCEP message. After reception of the TCP ACK which takes one RTT, the PCE sends a KeepAlive (line 3) to Gateway 1 and receives one (line 4). On reception of the KeepAlive by the PCE, Gateway 1 sends the path computation request. The PCE receives this one RTT after sending the KeepAlive (line 5). Subsequently, it triggers the establishment of a TCP and PCEP connection to Gateway 2 (10.4.0.99). In principle, the messages of the open sequence are exchanged asynchronously, i. e., there is no predefined order of messages. However, in our case, messages of the PCE trigger the according actions at Gateway 2 (lines 6-9). After sending the path computation request (line 10) and receiving the reply message (line 11), the PCE closes the PCEP connection to Gateway 2 (line 12). Finally, the PCE sends the reply to Gateway 1 (line 13).

For a subsequent request, the TCP as well as the PCEP connection between Gateway 1 and PCE is already established. When measuring the response time of such a subsequent multi-domain path request, the above mentioned jitter introduces some slight variations. On average, we obtain a response time of 273 ms. This overall response time includes 3 RTTs for TCP and PCEP connection setup between PCE and Gateway 2 and 2 RTTs for exchange of path requests and replies between Gateway 1, PCE and Gateway 2. The round trip times sum up to a value of about 260 ms. Thus the accumulated processing of all entities is in the order of 13 ms or even below.

6.2 Persistent Mode

The persistent operation only requires the TCP/PCEP handshakes for the very first request. All subsequent requests use established TCP and PCEP connections. Consequently, a trace analogue to Fig. 6 would not show lines 1-4, 6-9 and 12. Thus, we save 3 RTTs in contrast to the overall response time of the non-persistent mode. The measured average response time of about 117 ms reflects this consideration very well.

7 Conclusion

The contribution of this paper was three-fold. First, we studied both PCEP and DRAGON NARB architectures and provided a comprehensive comparison on their functionality and communication protocols. For the interconnection of both, we introduced an application layer gateway, which is able to translate one protocol to the other and vice versa. Second, we setup a multi-domain multi-layer testbed, which includes connection-oriented Ethernet as well as optical networks at two distant locations in Europe. For path computation, the domains exchange reachability information. We showed that multi-layer multi-domain path computation is feasible together with IETF PCE, DRAGON NARB and gateway implementations. Third, we evaluated the scenario and sampled the response time for the path computation for two different operations: persistent and non-persistent.

Our next steps are the improvement of the PCE implementation to serve requests in parallel as well as the enhancement of the gateway to support a larger number of IETF PCE functions, e. g., multiple requests per PCEP message. Besides, we are currently working towards a queuing theoretical model of the whole system.

Acknowledgement

The studies of this paper were partly carried out with the support of the BONE-project ("Building the Future Optical Network in Europe"), a Network of Excellence by the European Commission through the 7th ICT-Framework Programme and the Spanish national project MICINN DORADO TEC2009-07995. The authors would also thank Kristijan Kijurina for implementing the gateway.

References

1. Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., Swallow, G.: RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, IETF (December 2001)
2. Bianzino, A.P., Rougier, J.-L., Secci, S., Casellas, R., Martinez, R., Munoz, R., Djarallah, N.B., Douville, R., Pouyllau, H.: Testbed implementation of control plane extensions for inter-carrier gmpls lsp provisioning. In: International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities, pp. 1–6 (2009)
3. Casellas, R., Martinez, R., Munoz, R., Gunreben, S.: Enhanced Backwards Recursive Path Computation for Multi-area Wavelength Switched Optical Networks Under Wavelength Continuity Constraint. To Appear in the IEEE/OSA Journal of Optical Communications and Networking, A180–A193 (July 2009)
4. Cugini, F., Giorgetti, A., Andriolli, N., Paolucci, F., Valcarenghi, L., Castoldi, P.: Multiple path computation element (pce) cooperation for multi-layer traffic engineering. In: Optical Fiber Communication Conference (March 2007)
5. Farrel, A., Vasseur, J.-P., Ash, J.: A Path Computation Element (PCE)-Based Architecture. RFC 4655, IETF (August 2006)
6. Gunreben, S., Rambach, F.: Assessment and Performance Evaluation of PCE-based Inter-Layer Traffic Engineering. In: Proceedings of the IFIP Working Conference on Optical Network Design and Modelling (ONDM 2008) (March 2008)
7. IEEE Computer Society. 802.3: IEEE Standard for Local and Metropolitan Area Networks–Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications (2005)
8. ITU. Rec. Y.100: General overview of the Global Information Infrastructure standards development (1998)
9. ITU. Rec. Y.2011: General principles and general reference model for Next Generation Networks (October 2004)
10. Katz, D., Kompella, K., Yeung, D.: Traffic Engineering (TE) Extensions to OSPF Version 2. RFC 3630, IETF (September 2003)
11. Lehman, T., Sobieski, J., Jabbari, B.: Dragon: a framework for service provisioning in heterogeneous grid networks. IEEE Communications Magazine 44(3), 84–90 (2006)
12. Postel, J.: Transmission Control Protocol. RFC 793, IETF (September 1981)
13. Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description. RFC 3471, IETF (January 2003)
14. Path Computation Element (PCE) Communication Protocol (PCEP). RFC 5440, IETF (March 2009)
15. Yang, X.: Network Aware Resource Broker (NARB) Design and User Manual. Technical report, University of Southern California (USC), Information Sciences Institute (ISI) (June 2006)