Separation of the Control Plane and Forwarding Plane in Next-Generation Routers

Rachid Nait Takourout, Samuel Pierre, Laurent Marchand
Department of Computer Engineering, École Polytechnique de Montréal
C.P. 6079, succ. Centre-ville, Montréal, Qué., Canada H3C 3A7

Abstract: Ever since the Internet appeared, network elements, such as routers, have been enhanced with much functionality. They tend to increasingly integrate more services, such as support and security mechanisms, which ensure Quality of Service. Similarly, processor performance continues to increase rapidly. Thus, these two types of evolution trigger the concept of separating the Control plane and the Forwarding plane in network elements.This study addresses the separation of the control plane and the forwarding plane in Next-Generation Routers. XML syntax is used to model, on the one hand, the suitable elements of the control plane and the forwarding plane. Implementation and prototyping results indicate that this syntax type for modeling provides a flexible platform, which is truly able to manage the heterogeneity of the material implementing the Forwarding Plane.

Key words: Software-programmable router, ForCES, control plane, forwarding plane, TIPC

INTRODUCTION

Ever since the emergence of the Internet, network elements such as routers have been enhanced with many functionalities. They tend to increasingly integrate more services, such as support and security mechanisms to ensure Quality of Service. Similarly, processor performance continues to increase rapidly. Thus, these two types of evolution brought about the concept of separating the Control plane and the Forwarding plane in network elements. Moreover, the development of the new Network Processor technology which is able to treat packets while maintaining high performance levels and remaining programmable provides further support for this separation. However, this new networking area requires the development of a completely new architecture. Several organizations, such as the IETF, have shown an interest in this separation and a number of people are working to define the requirements and specifications of these incipient concepts.

This article presents a scalable and heterogeneously compliant solution to ensure the configuration management of elements within a network. This introduction presents the concepts related to network processor technology, the architecture proposed by ForCES, the IETF working group, Forwarding and Control Element Separation and some related technologies which offer potential solutions to manage the problem.

Network processor technology: The concept of separating the Control Plane and the Forwarding Plane could seem abstract and far from the routers' reality.

However, the evolution of routers offers concrete support for the separation. The two most represented router hardware architectures are, on the one hand, the software router based on a general purpose processor and on the other hand, the router based on ASIC processor (Application-Specific Integrated Circuit). Although, the first architecture provides a highly programmable structure, its performance remains inadequate. In fact, it is based on general purpose processors which are inadequate for network operations. Although the ASIC-based architecture provides a high level of performance when implementing network specific functionalities, its architecture remains expensive and minimally programmable.

The network processor architecture[1] has evolved from the aforementioned architectures. It is a compromise between the ASIC based and general purpose processor based architectures. In fact, a processor network is used to handle packets at wire speeds, while implementing a set of functions of Quality of Service, encryption, etc. The main attraction to this kind of processor comes from the implementation of these functions. Indeed, the processor network implements only part of the protocol stacks used in the node; only those which require direct data access and leave the more complex management to a standard processor. This type of processor has a material architecture characterized by:

* several processors, thus allowing the possibility of parallel processing;
* material entities specialized for network operations;
* fast memory access;

Corresponding Author: Samuel Pierre, Department of Computer Engineering, École Polytechnique de Montréal, C.P. 6079, succ. Centre-ville, Montréal, Qué., Canada H3C 3A7, Tel: (514) 340-4711 Ext. 4685, Fax: (514) 340-4658

* fast peripheral network access;
* general purpose processor access.

Since this technology is rather recent (the first marketed network processors became available in 1999), the lack of standards raises management and cohabitation problems for various types of network processors. More precisely, the need for flexibility in the evolution of the Control Plane and Forwarding Plane requires the creation of a true interface between these two planes which is powerful and able to manage the heterogeneity of the material implementing the Forwarding Plane.

ForCES Architecture: The ForCES working group proposes specific requirements which define a new scalable architecture in order to enhance the separation of the Control and the Forwarding Planes. It introduces logical entities presented in the draft IETF forces requirements[2] as:

* Forwarding Element (FE): A logical entity that implements the ForCES protocol. FEs use the underlying hardware to provide per-packet processing and handling as directed/controlled by a CE via the ForCES protocol.
* Control Element (CE): A logical entity that implements the ForCES protocol and uses it to instruct one or more FEs how to process packets. CEs handle functionality such as the execution of control and signaling protocols.
* Pre-Association Phase: The period when the management entities determine which FE and CE should be separated from the same network element.
* Post-Association Phase: The period when a CE knows how to control an FE and vice versa, including the CE-FE connection time.
* FE Manager: A logical entity that operates during the pre-association phase and is responsible for determining with which CE(s) an FE should communicate. An FE manager may use anything from static configuration to a pre-association phase protocol (see below) to determine which CE(s) to use. However this is currently out the ForCES working group's scope.
* CE Manager: A logical entity that operates during the pre-association phase and is responsible for determining with which FE(s) a CE should communicate.

Moreover, the Protocol Element (PE) term is relative to a CE or an FE. Figure 1 illustrates this architecture employing two CEs and two FEs.

Reliability and configuration problems: The main purpose of this study focuses on two elements. The first one concerns the reliability of the ForCES architecture. The second one deals with the configuration problem during the pre-association phase. This section addresses the first issue.
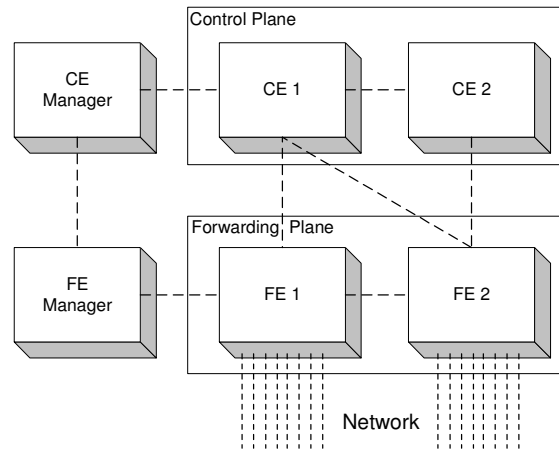

Fig. 1: Network element architecture example

According to the ForCES working group, the architecture reliability requirements are[3]:

* Scalability: The architecture must be able to support hundreds, at least, of FEs and tens of thousands of ports.
* Dynamical Association: The architecture must allow CEs and FEs to join and leave network elements dynamically.
* CE Redundancy: The architecture must support mechanisms for CE redundancy and CE failovers, including abilities to detect any loss of association.

Given these requirements, it becomes necessary to design a solution equipped with an underlying protocol to ensure reliable communication.

As for the second problem, notice that the ForCES architecture must provide a high compliance between CEs and FEs. In fact, they introduce the concept of the FE Model[4] that defines the inside behavior of an FE. This model is subdivided into two elements. The first one is the Capability Model that represents the packet manipulation as logical blocks. The second one, named the State Model, defines the FE's current configuration.

Separation related technologies: This section presents two technologies used to solve the problem: the Telecom Inter Process Communication protocol (TIPC)[5] and the eXtensible Markup Language (XML)[6].

TIPC is a message oriented communication service specially designed for cluster environments. It provides complete location transparency for intra-cluster messaging, using the concept of logical addressing with an internal address translation table. It is particularly suitable for systems where short, real-time critical transactions are performed. TIPC maintains a Quality of Service level which guarantees that messages cannot be lost or duplicated and that the message queue cannot be disturbed.

TIPC has the particularity of being easily portable. In fact, it defines its links to the network under a generic form called bearers that could be Ethernet,

ATM, InfiniBand amongst others. Moreover, bearers can be supported simultaneously and in multiple instances. This heterogeneous multi-homing property is further improved by employing a mechanism for transparent and disturbance-free link changeovers in case of bearer failure.

TIPC introduces the concept of having a port name to support logical addressing. The port name, an address composed of two 32-bit integers, is the permanent address used to establish a TIPC communication. The first integer, called 'type', identifies the type of service provided by a server and the second one, called 'instance', allows users to determine different instances of the same 'type'. This architecture provides a mechanism to partition the server's services by introducing the concept of a port name sequence that contains a 'type', both a lower and an upper instance. Figure 2 illustrates such address types. In fact, there are three TIPC servers: A, A' and B. For example, Server A publishes the port name sequence that indicates that it manages all of the clients who want to connect with 'type' 32456 and all instances between 0 and 99. In this case, Servers A and B partition the type of service represented by 'type' 32436. Moreover, Server A' publishes the same port name sequence as Server A. If the client connected to Server A detects a connection failure, it could try to reconnect directly using the same port name and in this case, it would be oriented to Server A'. This example illustrates the concept of logical addressing that is similar to CORBA mechanisms, although this type of mechanism is less flexible than the TIPC interface. In fact, TIPC uses a programming interface based on BSD sockets.
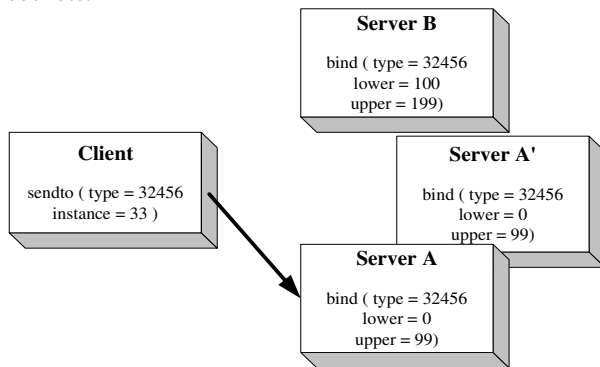

Fig. 2 : TIPC Client / Server Communication / Distributions

Before formalizing this model for CEs and FEs, consider the eXtensible Markup Language. XML is a markup language based on the Standard Generalized Markup Language (SGML) proposed by the World Wide Web Consortium (W3C) in 1996. The main purpose of XML is to dissociate documented data from its presentation. The markups of XML are extensible and one's own markup must be defined for each XML

document followed by an XML Schema or a Document Type Definition (DTD). To convert an XML document into a particular format, it must be parsed. There are two types of XML parsers. The first is one is based on treeing (based on the DOM API): it parses the document and creates a representative data structure. The second one is based on event mechanisms: it parses the document and launches specific callback functions for each markup. This type of parser uses the Simple API for XML (SAX).

Reliability problem: Upon reading the aforementioned ForCEs architectural requirements notice that robustness issues are closely related to the management of the redundancy of CEs and the dynamic management of PEs. As indicated in the introduction, the network processor technology could concretize the separation of the Control and the Forwarding Planes. In fact, FEs are logical representations of the functionalities of these processors. However, routers based on this technology are, in fact, traditional computers connected by a Backplane or local area network. This aspect enables users to place themselves within the framework of a computer cluster for the interconnection between CEs and FEs. By adopting this assumption, the correlation between the robustness requirements of ForCES architecture and the use of TIPC as the communication protocol becomes salient. The support of various communication technologies (UDP, ATM, Ethernet, etc.) and especially the logical addressing within a cluster makes the TIPC protocol an interesting solution when designing a robust architecture. Hence, this communication protocol will be used for the remainder of this article. This decision triggers a modification to certain aspects of the FACT protocol.

From a conceptual point of view, TIPC was used as protocol subjacent to the FACT protocol. Thus, each CE behaves as a TIPC server whose addressing (actually a port name) is inscribed on the entire cluster and each FE behaves symmetrically as a client. Using port names for addressing significantly simplifies the management of redundancy from the FE point of view. Each CE is associated with a given 'type' within the meaning of a port name, as well as a lower limit and higher threshold which are fixed for all of the port names published within the cluster. This choice is justified by the resulting addressing transparency.

Take, for example, a CE with a port name whose type is 17777. An FE becomes active and establishes a TIPC connection with this type. Then, the same operation is performed with another CE with the same port name. If an unspecified breakdown occurs during the first CE, the FE will be notified of the connection loss/failure by means of the TIPC protocol (when using TIPC with Ethernet for example, failures are detected within 1.5 seconds) and it will be able to attempt to reconnect itself with the same type 17777 as before. In this case, it will be detected automatically when it

communicates with the second CE. The use of TIPC as medium allows for the simpliication of a real architecture due to the redundancy management and the connection failure detection finds itself subjacent to the ForCES architecture. Moreover TIPC is a robust protocol, widely used over the past ten years. Although the FACT protocol ensures an explicit management of communication breakdowns through the use of timers, this type of mechanism is no longer necessary, as shown in the previous example, when TIPC is used. As FACT Heartbeat-type messages are no longer necessary, they will not be implemented.

The above example shows the contribution of TIPC with respect to the FACT protocol and the post-association phase. From the perspective of improving architecture robustness, it was decided to use TIPC for the pre-association phase. This choice was justified in two ways: the addressing event by the Managers and the configuration of the PE addresses. To manage the addressing of the Managers, the CE Managers and FE Managers are assigned two generic addresses known by all PEs. This enables PEs to become active with the ability to directly connect themselves to their respective Managers. In fact, the Managers will simply be the TIPC server and all of the PEs of the clients in order to simplify the pre-association phase and configuration. Moreover, this choice leads us to use the same redundancy process seen for CEs for the Managers in order to ensure the robustness of the pre-association phase. To illustrate this perspective, one can imagine that the first CE Manager becomes active and thus publishes the generic port name of the CE Managers. Then, a second CE Manager supplements the same step. The first will be responsible for managing all PE pre-associations. However, if the first one experiences a breakdown, the second to publish the same port name will already be able to ensure the role of managing all PE connections in a transparent and automatic way.

Although TIPC simplifies addressing, there remains an interrogation about the configuration of CE-FE pairings of the FACT protocol. As indicated by the ForCES working group, the configurations should be manual before a PE becomes active. Moreover, a network element must be able to manage hundreds of FEs and their configurations would thus quickly become problematic. However, TIPC offers a solution to the network element configuration problems. Indeed, the CE Managers will be assigned a new responsibility: the assignment of port names to all CEs. During the pre-association phase, the CE Managers will thus have to provide a port name for each new CE to ensure maximal coherence of the post-association phase.

Modeling and pre-association phase: Treating the modeling problem of the pre-association phase requires an explicit study of the requirements of this phase. To do so, it is necessary to first reconsider the function of the pre-association phase: pairing the CEs and FEs.

Indeed, the CE-FE comparison requires that these Managers know their functional structures. This section addresses the formalization of the CE-CE Manager and the FE-FE Manager and interaction by means of a representation in XML of the characteristics of each one. Let's examine the chronological evolution of the interaction between a CE and a CE Manager. This is a four-step process:

1. First, the CE signals its existence to the CE Manager. The CE Manager authenticates the CE and processes its request.
2. The CE Manager provides the CE with an identifier, called the CE-Id. Indeed, its attribution is not explicitly defined within the architectural framework suggested by the ForCES group.
3. The CE specifies its needs and restrictions (processing capacity, for example) to determine whether these services are available from FEs associated with the network elements and then receives the downstream from the CE Manager. However, the representation of its needs is particularly problematic. To solve this problem, IETF forces' requirements were further analyzed[2]. The types of switching functions implemented within an FE seem to indicate a simple declaration of the needs of CEs in this same format. Thus, it is assumed that a DTD can fulfill the needs (however, it is the CE which decides whether to work with the proposed FE or to violate the post-association phase mechanisms). However, it has its limitations such as the number of FEs each CE can manage. This model is called CE Specification.

Each DTD includes six major types of functional needs that can be expressed by the EC: (1) the port type that must be present, for example, an Ethernet interface; (2) the type of routing network, which can be an IP version 4 or 6; (3) the type of Quality of Service if the CE works within an IntServ or DiffServ architecture; (4) the type of encapsulation; (5) the type of security (in an access router, IPSec functions are necessary) and finally; (6) the type of functionalities provided by the manufacturers (only if FEs recognize them). To illustrate this approach, consider the example below which formalizes the needs of a CE in this XML document:

```
<CESpecification>
     <maxFE>3</maxFE>
     <EncapType>XML</ EncapType >
     <FailOver>Enable</ FailOver >
     <FunctionList size="2">
       <Function>
           <Port>
              <PortNb>2</PortNb>
              <PortTypeList size="1">
                 <PortType>Ethernet</PortType>
              </PortTypeList>
           </Port>
       </Function>
       <Function>
           <QoS>
```

```
<QoSTypeList size="1">
    <QoSType>DiffServ</QoSType>
    </QoSTypeList>
    </QoS>
    </Function>
    </FunctionList>
</CESpecification>
```

Note that this CE can only treat three FEs simultaneously, that it requires two Ethernet interfaces and that in addition, it uses DiffServ architecture for the border of a field. That makes it possible for the CE Manager to connect it to two FEs which can fulfill these requirements.

4. After this step, an event-based phase can be initiated. Indeed, once the three previous steps are supplemented, the CE Manager must know in which of three states the CE happens to be (i.e. active, inactive or standby state). Such information is thus exchanged transmitted via event messages to update the CE Manager. Two cases can be considered. The first corresponds simply to an inactivation of the CE for an unspecified reason during a normal operation. In this case, sending an asynchronous message suffices to ensure the follow-up with the CE Manager. The second case refers to a breakdown of the CE or a connection failure. The management of this kind of event can pass naturally through the use of periodic messages at a defined frequency. Given the specifications indicated in the section addressing the robustness of the ForCES protocol, this paper proposes the use of the TIPC protocol to detect this kind of event. This decomposition of the potential behavior of CE-CE Manager relations will be the base used while continuing to model the pre-association phase.

The same approach will be used for the interactions between FEs and the FE Manager. However, the modeling problem is different. In fact, the FE must provide its capacity to its FE Manager. In this case, an XML presentation of the FE Model proposed by the ForCES working group will be used. The capacity is represented by a list of Logical Block (LB) markups and the State Model by a list of markup Links:

```
<FEModel>
    <CapabilityModel>
        <LBList size="9" >
            <LB>...</LB>
        </LBList>
    </CapabilityModel>
    <StateModel>
        <LinkList size="9">
            <Link>...</Link>
        </LinkList>
    </StateModel>
</FEModel>
```

A handle and a type of functions are associated with each logical block. In this example, a DiffServ Meter:

```
<LB>
    <Handle>4</Handle>
    <Type>
        <DS_Meter>
            <MeterType>
                <Dir>In</Dir>
                <MeterCap>simpleTB</MeterCap>
                <HandleFail>5</HandleFail>
            </MeterType>
        </DS_Meter>
    </Type>
</LB>
```

The Link markup indicates the handles of a logical block that are directly connected:

```
<Link>
    <DownLBHandle>3</DownLBHandle>
    <UpLBHandle>8</UpLBHandle>
</Link>
```
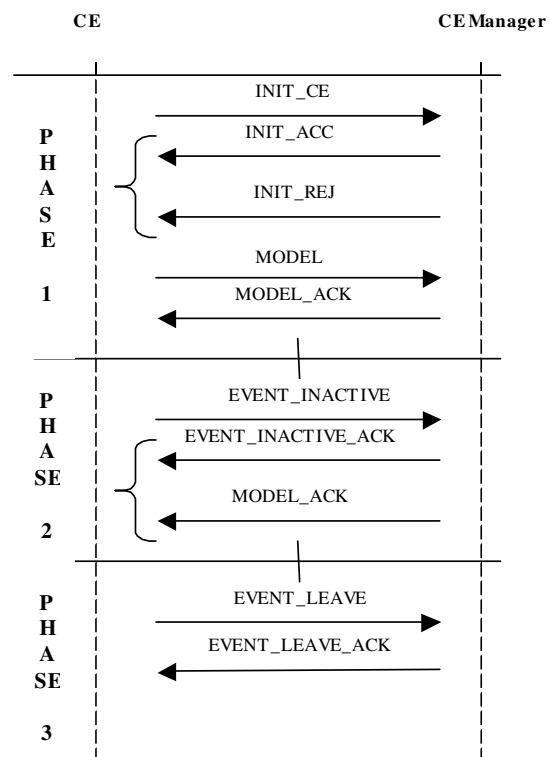

Fig. 3 : CE-CE Manager interactions

Figure 3 summarizes this new proposition. Of course, the same schema is obtained for the FE-FE Manager.

As for the interactions between the CE Manager and the FE Manager, a symmetric model whose event-based phase is composed of messages that contain the list of PE models is proposed. In fact, if an FE becomes inactive, the FE Manager has to send an updated list of FEs (and FE Models) to the CE Manager. This scenario is depicted in Fig. 4.

These diagrams illustrate the newly proposed protocol which is based on the header represented in Fig. 5.
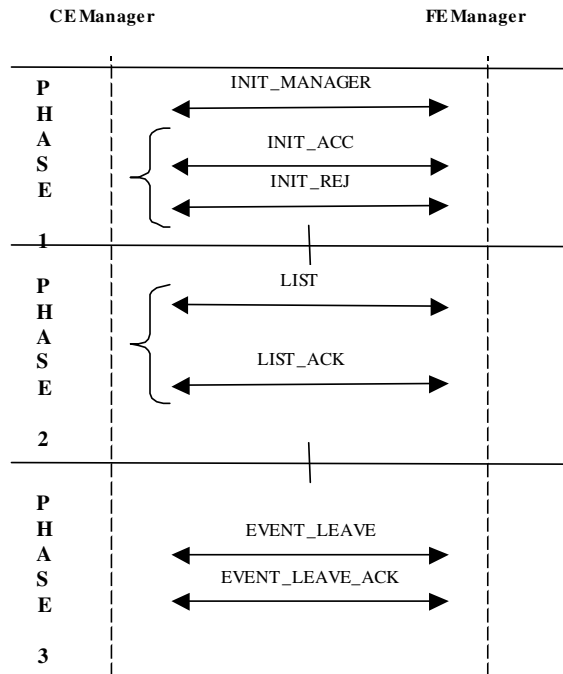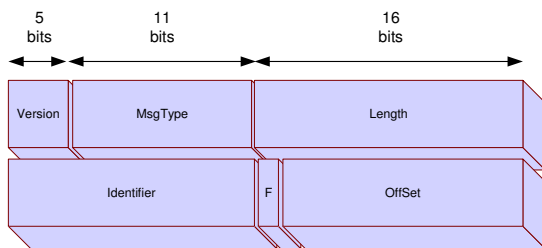
Fig. 4: CE Manager - FE Manager interactions

Fig. 5: PAP header

The pre-association protocol (PAP) header contains the type of message such as 'INIT_CE' in Fig. 3 and a fragmentation mechanism such as IPv4.

RESULTS

Proposed adaptation for implementation: The main purpose of this new proposal is to develop an interoperable and reliable architecture. For validation purposes, a simple prototype was used to represent the separation between the Control Plane and the Forwarding Plane. Hence, the architecture was modified in four ways to simplify the experiment:

* Structural Adaptation: The dichotomy between the CE Manager and the FE Manager corresponds to the finest granularity in the management of the PEs. However, in this implementation, it was decided to merge these entities.
* Modeling Adaptation: Only the Capability section of the FE Model was used to encapsulate the MODEL messages of the pre-association phase protocol. In fact, the state model is more specific than the post-association phase.
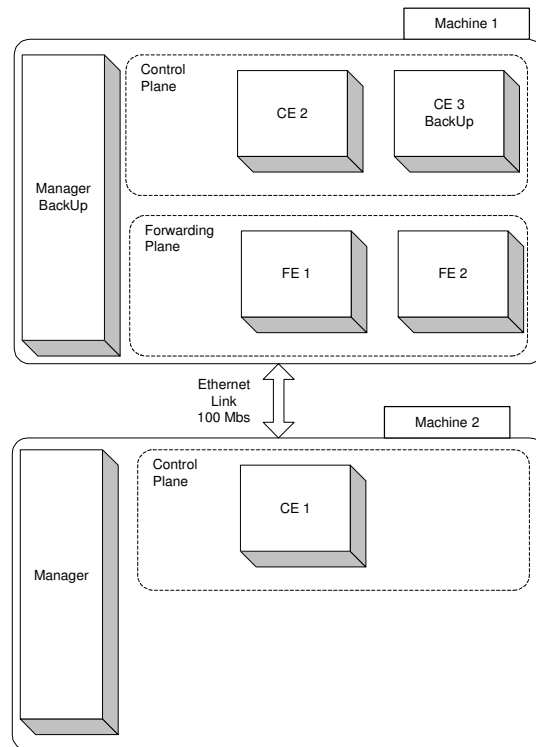

Fig. 6: Testing architecture

* Medium Adaptation: TIPC was selected as the communication medium for the entire architecture. Hence, each CE is a TIPC server thread and each FE is a TIPC client. The Manager entity is responsible for distributing the CE address inside the network element. Moreover, redundancy support was provided by giving the same address to two TIPC servers, so the Manager attributes an address to a CE in a judicious way in order to manage redundancy in the network element.
* Security Adaptation: To reach the objectives of this implementation, efforts were made to show the validity of these concepts. However, it is assumed that this work is conducted in a secure computer cluster and security problems are considered to be somewhat elusive. In this implementation however, in order to show that additional security mechanisms can easily be implemented, if, for example, the computer cluster becomes broader and requires more than one hop between machines. IPSec[7] technology was thus selected for the ease of its use as it acts upon the level three of OSI and is transparent for the roadbases. Although IPSec is hindered by its intrinsic complexity (using IKE), the ESP mechanism in Transport mode was selected for this implementation as it enables the use of a coding mechanism which is flexible with respect to the selection of the algorithm, but also due to its transparency compared to other platforms which are currently available.

Implementation environment and realization: In order to test this proposal, a two-machine architecture

was selected. Both machines use a Linux OS with a 2.4.18-3 kernel. To simulate the Control Plane and Forwarding Plane, a software platform developed by Intel© was used. Both machines were inter-connected via an Ethernet 100 Mb/s. Figure 6 presents this testing architecture.

In order to further specify the architecture, it is necessary to note that both computers were laptops. Machine 1 was equipped with a 733 MHz Celeron processor and Machine 2 a 500 MHz Pentium III. In fact, two types of tests were carried out. The first consisted of a functional test designed to assess the robustness and flexibility of this new architecture. The second series consisted of a benchmark of the architecture which was composed of three cases: Case 1, Case 2 and Case 3.

Functional tests: The purpose of the functional tests was to ensure that this architecture would support the stated functionalities of redundancy and survivability. Moreover, it is necessary to check that the management of CE addressing remains coherent from this approach. Thus, they were split up into two cases. The first is a study of the survivability of these concepts:

Sub-Case 1
Step 1: On Machine 2, the first Manager is launched. Then, on the same machine, a CE is launched. This CE's XML model indicates that it supports redundancy and requires Quality of DiffServ Service functionalities.
Step 2: On Machine 1, two CEs are launched. The first is the CE which manages redundancy and requires the same functionalities as the CE in Step 1. The second is a CE which requires IPv4 functionalities. Then, two FEs are launched in the same way. The first provides support for DiffServ conditioning functions whereas the second implements the IPv4 forwarder.
Step 3: A second Manager from Machine 1 is launched.
Step 4: Each FE sends a Join Request FACT message to connect itself to its CE.
Step 5: The Ethernet connection between the two machines is severed.

The second sub-case is designed to verify the platform behaviors in case an FE would not succeed in finding the CE where the connection must return to after the pre-association phase. In fact, it is a functional check of Step 2 of the pre-association protocol, so, only Machine 2 will be used to carry out the following steps:

Sub-Case 2
Step 1: A Manager is launched on Machine 2.
Step 2: The AFE implementing DiffServ presented in Step 2 Sub-Case 1 is launched.
Step 3: An FE is forced to try to send a Join Request FACT message.
Step 4: A CE whose XML modeling requires DiffServ Quality of Service functionalities is launched on Machine 2.

In the first sub-case, the Manager attributes an identifier and a port name to all CEs to launch their TIPC server. In this sub-case, what is most important is the notice that the Manager provides CE 1 and CE 3 with the same port name in order to ensure redundancy.

In fact, these CEs have the same CE Specifications. Another important point in this sub-case pertains to the moment where the connection between the two machines is severed: notice that the new Manager automatically starts up a new pre-association phase for all of the PE machines. The FE 1 that was connected to the CE 1, connects itself to the CE 3 which acts as a backup CE for CE 1.

Sub-case 2 shows that the event-based phase is entirely operational. In fact, the Manager receives the EVENT_INACTIVE message from the FE that does not have a CE and sends a new MODEL_ACK message when the corresponding CE becomes available.

Performance evaluation: The first case of these benchmark series must measure the time necessary to ensure the automatic reconnection of CEs and FEs in case of a breakdown between Machines 1 and 2. The second case was designed to assess the performance of this new architecture under various TIPC configurations. However, the most significant component of these tests aims to measure the impact of using IPSec to ensure the security of the architecture. The last case aims to measure the difference in performance between XML data encapsulation and the binary encapsulation with language C structures[8].

Case 1: This test included the structure of the functional test but only using the CE 1 and 3, FE 1 and both Managers. Although the preceding test illustrated the architecture, the required connection recovery time must be calculated.

First of all, communication was initiated between CEs and FEs and with the Manager. Then, the connection was severed just as with the functional test; however, for this test, a timer was used to calculate the time required to reconnect the CE 3 and FE 1 with the Safeguard Manager and to supplement the new pre-association phase. Then, the required time was increased for the FE to initiate a new Join Request type message for the CE 3 which is the safeguard CE for CE 1 and to receive the Join Response type message indicating that the connection has been effective.

The time required to conclude the pre-association phase is 65 milliseconds for the CE and 54 milliseconds for the FE. The connection recovery time for the FE with the safeguard CE equals 54 milliseconds. These results are illustrated in Fig. 7.

Note that the reconnection time for the FACT protocol is identical, whereas the time it took to conclude the phase starting from pre-association depends on the size of the XML documents modeled on each element, but also on the processing capacity of the machine on which the Manager is implemented.

Case 2: This test addresses the weight of the security connection between CEs and FEs. To supplement this test, once again, the functional CE 1 and FE 1 tests
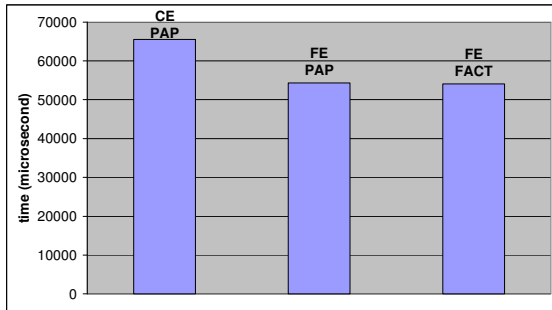
Fig. 7: Reconnection time for PAP and FACT protocols

Table 2: Add route performance with binary and XML data presentation

	Average Time (μs)	Add Route per Second
Binary Encapsulation	29592	92100
XML Encapsulation	216020	11605

were used. Then, the time required between the CE indication and reception of a delivery acknowledgement of the creation of a DiffServ structure used for the functional test was measured. To do so, this experiment was carried out 20 times, first using TIPC messages directly encapsulated on Ethernet screens, then UDP datagrams and finally with UDP and IPSec. The results of these experiments appear in Table 1.

Table 1: DiffServ structure: creation time

	Average Time (ms)	S.D.
TIPC over Ethernet	12	3
TIPC over UDP	970	260
TIPC over UDP & IPSec	1268	263

The analysis of these results shows a significant difference in the performance time between the use of Ethernet and UDP. This difference seems to be due to a dysfunction of the UDP adaptation layer of TIPC under Linux, since this problem does not arise with other operating systems. Moreover, this issue could be due to a bad implementation of the UDP stack in the Linux kernel 2.4.x although it should be corrected with the 2.6.x version. Regarding the UDP and UDP with IPSec performance discrepancy, note that there is a 30% decrease in performance. This result highlights the fact that additional security mechanisms increase the cost. Nevertheless, in this case, 30% is deemed acceptable.

Case 3:

This last benchmark test was designed to assess the impact of the selection of XML to model the data structure necessary for the FACT protocol. Hence, timers were used to compare the time required to send and receive a delivery acknowledgement for 500 Configure Logic Component type messages with each one encapsulating 50 add routes for a IPv4 routing table. In fact, a single machine was used. On this machine, the CE, an FE as well as a Manager were implemented. The results presented in Table 2 show the time required by these series of operations and presents the data in a binary way (with the C structures languages encapsulated directly in the messages) and in the XML structure.

It is clear that better ratio of add route per second was for the binary encapsulation. Just as for the addition of security functionalities, flexibility involves

compromising performance. Similar results were also obtained when a similar test on the number of add route per second in the form of XML documents was carried out, although two machines equipped with 2 GHz Pentium IV processors were used for this test. However, during this test, a ratio of 42000 add routes per second was obtained. This shows a clear correlation between the performance of the XML parser and the processing capabilities of the machines used.

CONCLUSION

The research objective of this paper was to design an evolutionary and flexible solution to separate the Control Plane and the Forwarding Plane. Hence, at first, a synthesis of the proposals made by the IETF was presented. The necessary analysis conducted by the ForCES work group yielded a focus on two significant issues: robustness and modeling. Thus, a combination of the combined use of the TIPC protocol as a communication support within the network element was proposed and XML was the language selected for data modeling and the pre- and post-association phases. It is significant that the analyses were conducted within a computer cluster. The choice of XML proved to be relevant when one considers the evolution of separation between the Control and the Forwarding Planes. The communication between these two planes is an essential concern for the future of this architecture. To validate this approach, a series of test cases were set up. These cases were simple yet they represented the importance of flexibility and robustness required by this separation well. The results obtained indicate that the functionalities of this architecture are conclusive. Moreover, two relevant concepts could be suggested to the ForCES working group: the first on the pre-association phase[9] and the second on XML modeling of FEs and CEs[10].

The main limitation observed in this study concerns the performance loss due to the use of the XML language to encode the data structures of the FACT protocol. First of all, note that the performance of the XML parsers is strongly coupled with the calculation capacities of the machines, but also that the flexibility options imply a reduction of performance. In addition, consider that in the architecture of the ForCES working group, the framework of the responsibilities of the CE and FE Managers are solely restricted to propose mailing lists to PEs. In this investigation, their responsibilities were more numerous but they were the only logical entities to have a global vision of the network element and could configure it in an optimal

way. However, FACT connections are left to the responsibility of the CEs which can either accept or refuse them. This fact reveals another limitation concerning the proposed architecture. It would be necessary to define the Managers responsibilities in the ForCES architecture in a more precisely to ensure enhanced cohesion in the pre-association protocol as well as for the remainder of the concepts.

ACKNOWLEDGEMENTS

REFERENCES

1. Spalink, T., S. Karlin, L. Peterson and Y. Gottlieb, 2001. Building a robust software-based router using network processors. ACM SIGOPS Operating Systems Review, Banff, Canada, 35: 216-229.

2. Khosravi, H. and T. Anderson, 2003. Requirements for separation of IP control and forwarding, draft-ietf-forces-requirements-10.txt.

3. Yang, L., R. Dantu, T. Anderson and R. Gopal, 2003. ForCES architectural framework, draft-ietf-forces-framework-06.txt.

4. Yang, L., J. Halpern, R. Gopal and A. DeKok, 2003. ForCES forwarding element functional model, draft-yang-forces-model-02.txt.

5. TIPC, http://tipc.sourceforge.net, August 2003.

6. Bray, T., J. Paoli, C.M. Sperberg-McQueen and E. Maler, 2000. Extensible Markup Language (XML) 1.0. Second Edn. http://www.w3.org/TR/REC-xml.

7. Kent, S. and R. Atkinson, 1998. Security architecture for the internet protocol. RFC 2402.

8. Network Processing Forum, http://www.npforum.org, August 2003

9. Nait, R.T., 2003. ForCES pre association phase protocol (PAP). draft-nait-forces-pap-00.txt.

10. Nait, R.T., 2003. ForCES XML based model for control and forward element separation. draft-nait-forces-xml-model-00.txt.