Internet Draft
Expiration: April 2003
File: draft-ietf-forces-requirements-07.txt
Working Group: ForCES

H. Khosravi,
T. Anderson (Editors)
Intel

October 2002

Requirements for Separation of IP Control and Forwarding

draft-ietf-forces-requirements-07.txt

Status of this Memo

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119].

1. Abstract

This document introduces the ForCES architecture and defines a set of associated terminology. This document also defines a set of architectural, modeling, and protocol requirements to logically separate the control and data forwarding planes of an IP (IPv4, IPv6, etc.) networking device.

2. Definitions

Addressable Entity (AE) - A physical device that is directly addressable given some interconnect technology. For example, on Ethernet, an AE is a device to which we can communicate using an Ethernet MAC address; on IP networks, it is a device to which we can communicate using an IP address; and on a switch fabric, it is a device to which we can communicate using a switch fabric port number.

Physical Forwarding Element (PFE) - An AE that includes hardware used to provide per-packet processing and handling. This hardware may consist of (but is not limited to) network processors, ASIC's, line cards with multiple chips or stand alone box with general-purpose processors.

PFE Partition - A logical partition of a PFE consisting of some subset of each of the resources (e.g., ports, memory, forwarding table entries) available on the PFE. This concept is analogous to that of the resources assigned to a virtual switching element as

described in [REQ-PART].

Physical Control Element (PCE) - An AE that includes hardware used to provide control functionality. This hardware typically includes a general-purpose processor.

PCE Partition - A logical partition of a PCE consisting of some subset of each of the resources available on the PCE.

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 by a CE via the ForCES protocol. FEs may use PFE partitions, whole PFEs, or multiple PFEs.

Proxy FE - A name for a type of FE that cannot directly modify its underlying hardware but instead manipulates that hardware using some intermediate form of communication (e.g., a non-ForCES protocol or DMA). A proxy FE will typically be used in the case where a PFE cannot implement the ForCES protocol directly (e.g., due to the lack of a general purpose CPU).

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. CEs may consist of PCE partitions or whole PCEs.

Pre-association Phase - The period of time during which a FE Manager (see below) and a CE Manager (see below) are determining which FE and CE should be part of the same network element. Any partitioning of PFEs and PCEs occurs during this phase.

Post-association Phase - The period of time during which a FE does know which CE is to control it and vice versa, including the time during which the CE and FE are establishing communication with one another.

ForCES Protocol - While there may be multiple protocols used within the overall ForCES architecture, the term "ForCES protocol" refers only to the ForCES post-association phase protocol (see below).

ForCES Post-Association Phase Protocol - The protocol used for post-association phase communication between CEs and FEs. This protocol

does not apply to CE-to-CE communication, FE-to-FE communication, or to communication between FE and CE managers. The ForCES protocol is a master-slave protocol in which FEs are slaves and CEs are masters. This protocol includes both the management of the communication channel (e.g., connection establishment, heartbeats) and the control messages themselves. This protocol could be a single protocol or could consist of multiple protocols working together.

FE Model - A model that describes the logical processing functions of a FE.

FE Manager - A logical entity that operates in the pre-association phase and is responsible for determining to which CE(s) a FE should communicate. This process is called CE discovery and may involve the FE manager learning the capabilities of available CEs. A FE

manager may use anything from a static configuration to a pre-association phase protocol (see below) to determine which CE(s) to use. Being a logical entity, a FE manager might be physically combined with any of the other logical entities mentioned in this section.

CE Manager - A logical entity that operates in the pre-association phase and is responsible for determining to which FE(s) a CE should communicate. This process is called FE discovery and may involve the CE manager learning the capabilities of available FEs. A CE manager may use anything from a static configuration to a pre-association phase protocol (see below) to determine which FE to use. Being a logical entity, a CE manager might be physically combined with any of the other logical entities mentioned in this section.

Pre-association Phase Protocol - A protocol between FE managers and CE managers that is used to determine which CEs or FEs to use. A pre-association phase protocol may include a CE and/or FE capability discovery mechanism. Note that this capability discovery process is wholly separate from (and does not replace) that used within the ForCES protocol (see Section 7, requirement #1). However, the two capability discovery mechanisms may utilize the same FE model (see Section 6). Pre-association phase protocols are not discussed further in this document.

ForCES Network Element (NE) - An entity composed of one or more CEs and one or more FEs. To entities outside a NE, the NE represents a

single point of management. Similarly, a NE usually hides its internal organization from external entities.

ForCES Protocol Element - A FE or CE.

High Touch Capability - This term will be used to apply to the capabilities found in some forwarders to take action on the contents or headers of a packet based on content other than what is found in the IP header. Examples of these capabilities include NAT-PT, firewall, and L7 content recognition.

3. Introduction

An IP network element is composed of numerous logically separate entities that cooperate to provide a given functionality (such as a routing or IP switching) and yet appear as a normal integrated network element to external entities. Two primary types of network element components exist: control-plane components and forwarding-plane components. In general, forwarding-plane components are ASIC, network-processor, or general-purpose processor-based devices that handle all data path operations. Conversely, control-plane components are typically based on general-purpose processors that

provide control functionality such as the processing of routing or signaling protocols. A standard set of mechanisms for connecting these components provides increased scalability and allows the control and forwarding planes to evolve independently, thus promoting faster innovation.

For the purpose of illustration, let us consider the architecture of a router to illustrate the concept of separate control and forwarding planes. The architecture of a router is composed of two main parts. These components, while inter-related, perform functions that are largely independent of each other. At the bottom is the forwarding path that operates in the data-forwarding plane and is responsible for per-packet processing and forwarding. Above the forwarding plane is the network operating system that is responsible for operations in the control plane. In the case of a router or switch, the network operating system runs routing, signaling and control protocols (e.g., RIP, OSPF and RSVP) and dictates the forwarding behavior by manipulating forwarding tables, per-flow QoS tables and access control lists. Typically, the architecture of these devices combines all of this functionality into a single functional whole with respect to external entities.

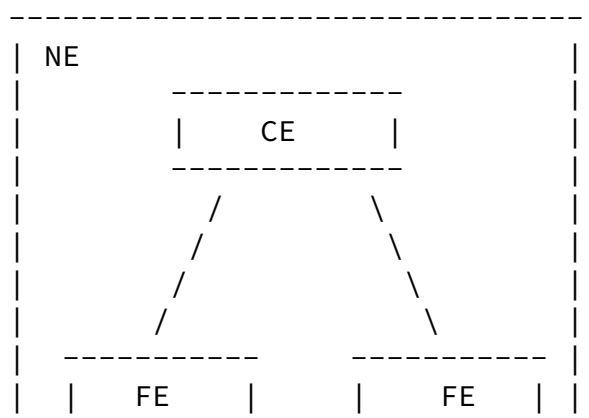
4. Architecture

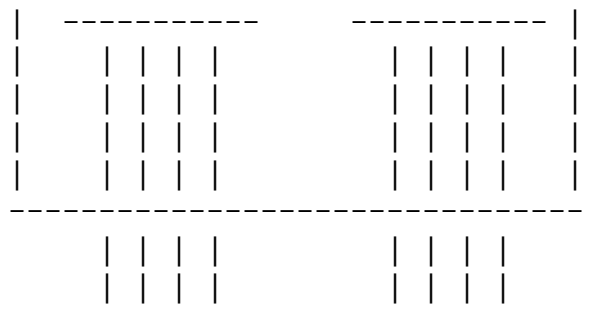
The chief components of a NE architecture are the CE, the FE, and the interconnect protocol. The CE is responsible for operations such as signaling and control protocol processing and the implementation of management protocols. Based on the information acquired through control processing, the CE(s) dictates the packet-forwarding behavior of the FE(s) via the interconnect protocol. For example, the CE might control a FE by manipulating its forwarding tables, the state of its interfaces, or by adding or removing a NAT binding.

The FE operates in the forwarding plane and is responsible for per-packet processing and handling. By allowing the control and forwarding planes to evolve independently, different types of FEs can be developed - some general purpose and others more specialized. Some functions that FEs could perform include layer 3 forwarding, metering, shaping, firewall, NAT, encapsulation (e.g., tunneling), decapsulation, encryption, accounting, etc. Nearly all combinations of these functions may be present in practical FEs.

Below is a diagram illustrating an example NE composed of a CE and two FEs. Both FEs and CE require minimal configuration as part of the pre-configuration process and this may be done by FE Manager and CE Manager respectively. Apart from this, there is no defined role for FE Manager and CE Manager. The Proxy FE has also been defined for clarification purpose. These components are out of scope of the

architecture and requirements for the ForCES protocol, which only involves CEs and FEs.





5. Architectural Requirements

The following are the architectural requirements:

1) CEs and FEs MUST be able to connect by a variety of interconnect technologies. Examples of interconnect technologies used in current architectures include Ethernet,bus backplanes, and ATM (cell) fabrics. FEs MAY be connected to each other via a different technology than that used for CE/FE communication.

2) FEs MUST support a minimal set of capabilities necessary for establishing network connectivity (e.g., interface discovery, port up/down functions). Beyond this minimal set, the ForCES architecture MUST NOT restrict the types or numbers of capabilities that FEs may contain.

3) Packets MUST be able to arrive at the NE by one FE and leave the NE via a different FE.

4) A NE MUST support the appearance of a single functional device. For example, in a router, the TTL of the packet should be decremented only once as it traverses the NE regardless of how many FEs through which it passes. However, external entities (e.g., FE managers and CE managers) MAY have direct access to individual ForCES protocol elements for providing information to transition them from the pre-association to post-association phase.

5) The architecture MUST provide a way to prevent unauthorized ForCES protocol elements from joining a NE.(For more protocol details, refer to section 7 requirement# 2)

6) A FE MUST be able to asynchronously inform the CE of a failure or increase/decrease in available resources or capabilities on the FE.

Thus the FE MUST support error monitoring and reporting. (Since there is not a strict 1-to-1 mapping between FEs and PFEs, it is possible for the relationship between a FE and its physical resources to change over time). For example, the number of physical ports or the amount of memory allocated to a FE may vary over time. The CE needs to be informed of such changes so that it can control the FE in an accurate way.

7) The architecture MUST support mechanisms for CE redundancy or CE failover. This includes the ability for CEs and FEs to determine when there is a loss of association between them, ability to restore association and efficient state (re)synchronization mechanisms. This also includes the ability to preset the actions an FE will take in reaction to loss of association to its CE e.g., whether the FE will continue to forward packets or whether it will halt operations.

8) FEs MUST be able to redirect control packets (such as RIP, OSPF messages) addressed to their interfaces to the CE. They MUST also redirect other relevant packets (e.g., such as those with Router Alert Option set) to their CE. The CEs MUST be able to configure the packet redirection information/filters on the FEs. The CEs MUST also be able to create packets and have its FEs deliver them.

9) Any proposed ForCES architectures MUST explain how that architecture supports all of the router functions as defined in [RFC1812].

10) In a ForCES NE, the FEs MUST be able to provide their topology information (topology by which the FEs in the NE are connected) to the CE(s).

11) The ForCES NE architecture MUST be capable of supporting (i.e., must scale to) at least hundreds of FEs and tens of thousands of ports.

12) The ForCES architecture MUST allow FEs AND CEs to join and leave NEs dynamically.

13) The ForCES NE architecture MUST support multiple CEs and FEs.

14) For pre-association phase setup, monitoring, configuration issues, it MAY be useful to use standard management mechanisms for

CEs and FEs. The ForCES architecture and requirements do not preclude this. In general, for post-association phase, most management tasks SHOULD be done through interaction with the CE. In certain conditions (e.g. CE/FE disconnection), it may be useful to allow management tools (e.g. SNMP) to be used to diagnose and repair problems. The following guidelines MUST be observed:

1. The ability for a management tool (e.g. SNMP) to be used to read (but not change) the state of FE SHOULD NOT be precluded.
2. It MUST NOT be possible for management tools (e.g. SNMP, etc) to change the state of a FE in a manner that affects overall NE behavior without the CE being notified.

6. FE Model Requirements

The variety of FE functionality that the ForCES architecture allows poses a potential problem for CEs. In order for a CE to effectively control a FE, the CE must understand how the FE processes packets. We therefore REQUIRE that a FE model be created that can express the logical packet processing capabilities of a FE. This model will be used in the ForCES protocol to describe FE capabilities (see Section 7, requirement #1). The FE model MUST define both a capability model and a state model, which expresses the current configuration of the device. The FE model MUST also support multiple FEs in the NE architecture.

6.1. Types of Logical Functions

The FE model MUST express what logical functions can be applied to packets as they pass through a FE.
Logical functions are the packet processing functions that are applied to the packets as they are forwarded through a FE. Examples of logical functions are layer 3 forwarding, firewall, NAT, shaping. Section 6.5 defines the minimal set of logical functions that the FE Model MUST support.

6.2. Variations of Logical Functions

The FE model MUST be capable of supporting/allowing variations in the way logical functions are implemented on a FE. For example, on a certain FE the forwarding logical function might have information about both the next hop IP address and the next hop MAC address, while on another FE these might be implemented as separate logical functions. Another example would be NAT functionality that can have several flavors such as Traditional/Outbound NAT, Bi-directional NAT, Twice NAT, Multihomed NAT [RFC2663]. The model must be flexible enough to allow such variations in functions.

6.3. Ordering of Logical Functions

The model MUST be capable of describing the order in which these logical functions are applied in a FE. The ordering of logical

functions is important in many cases. For example, a NAT function may change a packet's source or destination IP address. Any number of other logical functions (e.g., layer 3 forwarding, ingress/egress firewall, shaping, accounting) may make use of the source or destination IP address when making decisions. The CE needs to know whether to configure these logical functions with the pre-NAT or post-NAT IP address. Furthermore, the model MUST be capable of expressing multiple instances of the same logical function in a FE's processing path. Using NAT again as an example, one NAT function is typically performed before the forwarding decision (packets arriving externally have their public addresses replaced with private addresses) and one NAT function is performed after the forwarding decision (for packets exiting the domain, their private addresses are replaced by public ones).

6.4. Flexibility

Finally, the FE model SHOULD provide a flexible infrastructure in which new logical functions and new classification, action, and parameterization data can be easily added. In addition, the FE model MUST be capable of describing the types of statistics gathered by each logical function.

6.5. Minimal Set of Logical Functions

The rest of this section defines a minimal set of logical functions that any FE model MUST support. This minimal set DOES NOT imply that all FEs must provide this functionality. Instead, these requirements only specify that the model must be capable of expressing the capabilities that FEs may choose to provide.

1)Port Functions
The FE model MUST be capable of expressing the number of ports on the device, the static attributes of each port (e.g., port type, link speed), and the configurable attributes of each port (e.g., IP address, administrative status).

2)Forwarding Functions
The FE model MUST be capable of expressing the data that can be used by the forwarding function to make a forwarding decision. Support for IPv4 and IPv6 unicast and multicast forwarding functions MUST be provided by the model.

3)QoS Functions
The FE model MUST allow a FE to express its QoS capabilities in

terms of, e.g., metering, policing, shaping, and queuing functions. The FE model MUST be capable of expressing the use of these functions to provide IntServ or DiffServ functionality as described in [RFC2211], [RFC2212], [RFC2215], [RFC2475], and [RFC3290].

4)Generic Filtering Functions
The FE model MUST be capable of expressing complex sets of filtering functions. The model MUST be able to express the existence of these functions at arbitrary points in the sequence of a FE's packet processing functions. The FE model MUST be capable of expressing a wide range of classification abilities from single fields (e.g., destination address) to arbitrary n-tuples. Similarly, the FE model MUST be capable of expressing what actions these filtering functions can perform on packets that the classifier matches.

5)Vendor-Specific Functions
The FE model SHOULD be extensible so that new, currently unknown FE functionality can be expressed. The FE Model SHOULD NOT be extended to express standard/common functions in a proprietary manner. This would NOT be ForCES compliant.

6)High-Touch Functions
The FE model MUST be capable of expressing the encapsulation and tunneling capabilities of a FE. The FE model MUST support functions that mark the class of service that a packet should receive (i.e. IPv4 header TOS octet or the IPv6 Traffic Class octet). The FE model MAY support other high touch functions (e.g., NAT, ALG).

7)Security Functions
The FE model MUST be capable of expressing the types of encryption that may be applied to packets in the forwarding path.

8)Off-loaded Functions
Per-packet processing can leave state in the FE, so that logical functions executed during packet processing can perform in a consistent manner (for instance, each packet may update the state of the token bucket occupancy of a give policer). In addition, FEs MUST allow logical functions to execute asynchronously from packet processing, according to a certain finite-state machine, in order to perform functions that are, for instance, off-loaded from the CE to the FE. The FE model MUST be capable of expressing these asynchronous functions. Examples of such functions include the finite-state machine execution required by TCP termination or OSPF

Hello processing, triggered not only by packet events, but by timer events as well.

7. ForCES Protocol Requirements

This section specifies some of the requirements that the ForCES protocol MUST meet.

1)Configuration of Modeled Elements

The ForCES protocol MUST allow the CEs to determine the capabilities of each FE. These capabilities SHALL be expressed using the FE model whose requirements are defined in Section 6. Furthermore, the protocol MUST provide a means for the CEs to control all the FE capabilities that are discovered through the FE model. The protocol MUST be able to add/remove classification/action entries, set/delete parameters, query statistics, and register for and receive events.

2)Support for Secure Communication

a) FE configuration will contain information critical to the functioning of a network (e.g. IP Forwarding Tables). As such, it MUST be possible to ensure the authenticity and integrity of ForCES protocol messages.
b) FE configuration information may also contain information derived from business relationships (e.g. service level agreements). Therefore, it MUST be possible to secure (make private) ForCES protocol messages.
c) In order to provide authentication, integrity and privacy for a proposed ForCES protocol, existing security mechanisms such as IPSec/IKE, TLS, etc. MUST be used/leveraged if applicable.

3)Scalability

The ForCES protocol MUST be capable of supporting (i.e., must scale to) at least hundreds of FEs and tens of thousands of ports. For example, the ForCES protocol field sizes corresponding to FE or port numbers SHALL be large enough to support the minimum required numbers. This requirement does not relate to the performance of a NE as the number of FEs or ports in the NE grows.

4)Multihop

When the CEs and FEs are separated beyond a single hop, the ForCES protocol will make use of an existing RFC2914 compliant L4 protocol

with adequate reliability, security and congestion control (e.g. TCP, SCTP) for transport purposes.

5)Message Priority
The ForCES protocol MUST provide a means to express the protocol message priorities.

6)Reliability
The ForCES protocol SHALL assume that it runs on top of an unreliable, datagram service. For IP networks, an encapsulation of the ForCES protocol SHALL be defined that uses a [RFC2914]-compliant transport protocol and provides a datagram service (that could be unreliable).
For non-IP networks such as FastEth, GigE, cell fabric between control and forwarding elements, additional encapsulations MAY be defined so long as they provide a datagram service to the ForCES protocol.

However, reliability in the ForCES context means the ability to acknowledge the content and execution of messages, rather than just their delivery. For this reason, the ForCES protocol MUST provide internal support for reliability mechanisms such as message acknowledgements and/or state change confirmations. This is required for both IP and non-IP networks.

7)Interconnect Independence
The ForCES protocol MUST support a variety of interconnect technologies. (refer to section 5, requirement# 1)

8)CE redundancy or CE failover
The ForCES protocol MUST support mechanisms for CE redundancy or CE failover. This includes the ability for CEs and FEs to determine when there is a loss of association between them, ability to restore association and efficient state (re)synchronization mechanisms. This also includes the ability to preset the actions an FE will take in reaction to loss of association to its CE e.g., whether the FE will continue to forward packets or whether it will halt operations. (refer to section 5, requirement# 7)

9)Packet Redirection
The ForCES protocol MUST define a way for FEs to be able to redirect control packets (such as RIP, OSPF messages) addressed to their interfaces to the CE and vice-versa. It MUST also allow redirection

of other relevant packets (e.g., such as those with Router Alert Option set) to the CE. The ForCES protocol MUST also allow CEs to configure the packet redirection information/filters on the FEs. (refer to section 5, requirement# 8)

10)Topology Exchange
The ForCES protocol MUST allow the FEs to provide their topology information (topology by which the FEs in the NE are connected) to the CE(s). (refer to section 5, requirement# 10)

11)Dynamic Association
The ForCES protocol MUST allow CEs and FEs to join and leave a NE dynamically. (refer to section 5, requirement# 12)

12)Command Bundling
The ForCES protocol MUST be able to group an ordered set of commands to a FE. Each such group of commands SHOULD be sent to the FE in as few messages as possible. Furthermore, the protocol MUST support the ability to specify if a command group MUST have all-or-nothing semantics.

13)Asynchronous Event Notification

The ForCES protocol MUST be able to asynchronously notify the CE of events on the FE such as failures or change in available resources or capabilities. (refer to section 5, requirement# 6)

14)Query Statistics
The ForCES protocol MUST provide a means for the CE to be able to query statistics (monitor performance) from the FE.

8. References

8.1.Normative References

[RFC3290] Y. Bernet, et. al., "An Informal Management Model for DiffServ Routers", , May 2002.

[RFC1812] F. Baker, "Requirements for IP Version 4 Routers", RFC1812, June 1995.

[RFC2211] J. Wroclawski, "Specification of the Controlled-Load

Network Element Service", RFC2211, September 1997.

[RFC2212] S. Shenker, C. Partridge, R. Guerin, "Specification of Guaranteed Quality of Service", RFC2212, September 1997.

[RFC2215] S. Shenker, J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", RFC2215, September 1997.

[RFC2475] S. Blake, et. Al., "An Architecture for Differentiated Service", RFC2475, December 1998.

[RFC2914] S. Floyd, "Congestion Control Principles", RFC2914, September 2000.

[RFC2663] P. Srisuresh & M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC2663, August 1999.

8.2.Informative References

[REQ-PART] T. Anderson, J. Buerkle, "Requirements for the Dynamic Partitioning of Switching Elements", work in progress, July 2002, <draft-ietf-gsmp-dyn-part-reqs-02.txt>.

9. Security Considerations

See architecture requirement #5 and protocol requirement #2.

10. Authors' Addresses & Acknowledgments

This document was written by the ForCES Requirements design team:

Todd A. Anderson (Editor)

Ed Bowen
IBM Zurich Research Laboratory
Saumerstrasse 4
CH-8803 Rueschlikon Switzerland
Phone: +41 1 724 83 68
Email: edbowen@us.ibm.com

Ram Dantu
Netrake Corporation
3000 Technology Drive, #100,
Plano, Texas, 75074
Email: rdantu@netrake.com
Phone: 214 291 1111

Avri Doria
Institute for System Technology
Lulea University of Technology
SE-971 87, Lulea, Sweden
Phone: +46 (0)920 49 3030
Email: avri@sm.luth.se

Ram Gopal
Nokia Research Center
5, Wayside Road,
Burlington, MA 01803
Phone: 1-781-993-3685
Email: ram.gopal@nokia.com

Jamal Hadi Salim
Znyx Networks
Ottawa, Ontario
Canada
Email: hadi@znyx.com

Hormuzd Khosravi (Editor)

Muneyb Minhazuddin
Avaya Inc.
123, Epping road,
North Ryde, NSW 2113, Australia
Phone: +61 2 9352 8620
email: muneyb@avaya.com

Margaret Wasserman

Wind River
10 Tara Blvd., Suite 330
Nashua, NH 03062
Phone: +1 603 897 2067
Email: mrw@windriver.com

The authors would like to thank Vip Sharma and Lily Yang for their valuable contributions.

11. Editors' Contact Information

Hormuzd Khosravi
Intel
2111 NE 25th Avenue
Hillsboro, OR 97124 USA
Phone: +1 503 264 0334
Email: hormuzd.m.khosravi@intel.com

Todd A. Anderson
Intel
2111 NE 25th Avenue
Hillsboro, OR 97124 USA
Phone: +1 503 712 1760
Email: todd.a.anderson@intel.com