Internet Draft
Expiration: May 2002
File: draft-ietf-forces-requirements-01.txt
Working Group: ForCES

T. Anderson
Intel Labs
E. Bowen
IBM
R. Dantu
Netrake Inc.
A. Doria
N/A
J. Hadi Salim
Znyx Networks
H. Khosravi
Intel Labs
M. Minhazuddin
Avaya Inc.
M. Wasserman
Wind River
November 2001

Requirements for Separation of IP Control and Forwarding

draft-ietf-forces-requirements-01.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 presents an introduction to issues surrounding a ForCES architecture and defines a set of associated terminology. Subsequently, this document defines a set of architectural, modeling, and protocol requirements for mechanisms to logically separate the control and data forwarding planes of an IP network 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, or general-purpose processors. For example, line cards in a forwarding backplane are PFEs.

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 router [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 (e.g., due to the lack of a general purpose CPU) the ForCES protocol directly.

Control Element (CE) - A logical entity that implements the ForCES protocol and uses it to instruct one or more FEs as to how they should process packets. CEs handle functionality such as the execution of control and signaling protocols. CEs may encompass PCE partitions or whole PCEs.

Pre-association Phase - The period of time during which a FE Manager (see definition below) and a CE Manager (see definition below) are deciding which FE and CE should be part of the same network element.

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 (after they have been associated to the same NE).

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.

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 determination 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 determination 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 helps them determine which CEs or FEs are to be associated to a NE. A pre-association phase protocol may include a CE and/or FE capability discovery mechanism. It is important to 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 separated 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 would provide increased scalability and allow 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 and value of separate control and

forwarding planes. The architecture of a router is composed of two

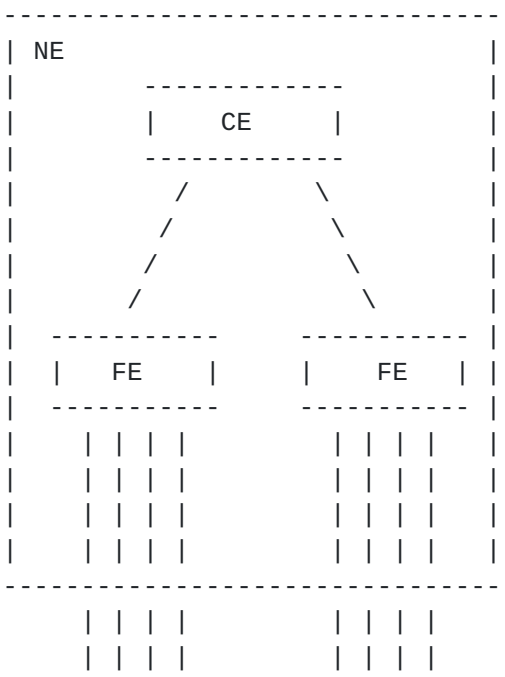
main parts. These components, while inter-related, perform functions that are largely independent of each other. At the bottom is the forwarding path that operates in the data-forwarding plane and is responsible for per-packet processing and forwarding. Above the forwarding plane is the network operating 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 mainly 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 its 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 chiefly for per-packet processing and handling. By allowing the control and forwarding planes to evolve independently, we expect different types of FEs to 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.



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 connections, 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.

6) A FE MUST be able to asynchronously inform the CE of an increase/decrease in available resources or capabilities on the FE. (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) CEs and FEs MUST determine when a loss of connectivity between them has occurred.

8) FEs MUST redirect packets addressed to their interfaces to their CE for further processing. Furthermore, FEs MUST redirect other required packets (e.g., such as those with the router alert option set) to their CE as well. (FEs MAY provide any other classification/redirection capabilities that they desire as described in Section 6.4 requirement #4.) Similarly, CEs MUST be able to create packets and have its FEs deliver them.

9) All proposed ForCES architectures MUST explain how that architecture may be applied to support all of a router's functions as defined in [RFC1812].

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

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) FEs MUST be able to join and leave NEs dynamically.

13) CEs MUST be able to join and leave NEs dynamically.

14) The NE architecture MUST support multiple CEs and FEs.

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 at a logical level 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).

6.1. Higher-Level FE Model

At its higher level, the FE model MUST express what logical functions can be applied to packets as they pass through a FE. Furthermore, the model MUST be capable of describing the order in which these logical functions are applied in a FE. This ordering 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).

6.2. Lower-Level FE Model

At its lower level, the FE model MUST be able to express the capabilities of each logical function. As the following examples will illustrate, these lower-level capabilities come in five varieties: classification, action, parameterization, statistic, and events.

6.2.1. Classification

Classification data is used by a logical function to perform pattern matching. For example, there may be two FEs, both of which provide an ingress firewall function. However, the first FE may filter on any subset of a (source IP, destination IP, IP protocol, source port, destination port)-tuple whereas the second FE may perform only a (destination IP, IP protocol)-tuple classification. In either case, the action applied to matching packets may be the same, e.g. drop.

6.2.2. Action

Action data is used by a logical function to manipulate the packet because of a classification. For example, there may be two traffic-shaping functions, both of which classify only on destination IP address. However, one of the shaping functions may implement a leaky bucket that requires two parameters whereas the other function

may implement a token bucket that requires three parameters.

6.2.3. Parameterization

Parameterization data can be viewed as parameters to the logical function itself. This data does not affect individual packets but affects how the function as a whole behaves. For example, there may be a congestion function that implements two varieties of RED that require different parameter sets. Parameterization data would be used to select between the two varieties of RED and to provide the necessary configuration to each variety.

6.2.4.Statistics

Some logical functions may maintain certain statistics (e.g., number of packets processed) about their own operation. The FE model needs to express which statistics a FEÆs logical functions maintain so that CEs can later query those statistics to answer queries made by higher level entities.

6.2.5.Event

Some logical functions may be able to generate asynchronous events that can be sent to the control plane. Two examples of these events include packet redirection events and port state change events (e.g., up/down). The FE model must be able to express the events that a FEÆs logical functions may generate so that CEs can register to receive those events when they happen to occur.

6.3. 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. Also, the FE model MUST be capable of describing the types of statistics gathered by each logical function.

6.4. Minimal Set of Logical Functions

The rest of this section defines a minimal set of logical functions that any FE model MUST support. However, this section shall not be construed as to define a set of functions that all FEs must provide. On the contrary, FEs are not required to support any of the following functions. These requirements only specify that the FE model must be capable of expressing the capabilities that FEs are likely to initially 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.

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], and [DS-PIB].

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 multiples of these functions at arbitrary points in a FE's packet processing path. 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 vendor-specific functionality can be expressed.

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 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.

7. ForCES Protocol Requirements

This section specifies some of the requirements that a 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. (For example, 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
Since FE configuration will contain information critical to the functioning of a network (such as IP forwarding tables) and may contain information derived from business relationships (e.g., SLAs), the ForCES protocol MUST support a method of securing communication between FEs and CEs to ensure that information is delivered privately and in an unmodified form.

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 the protocol 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 message priority.

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, additional encapsulations MAY be defined so long as they provide a datagram service to the ForCES protocol. However, since some messages will need to be reliably delivered to FEs, the ForCES protocol MUST provide internal support for reliability mechanisms such as message acknowledgements and/or state change confirmations.

8. Security Considerations

See architecture requirement #5 and protocol requirement #2.

9. References

[DS-PIB] M. Fine, et. al., "Differentiated Services Quality of Service Policy Information Base", work in progress, November 2001, <draft-ietf-diffserv-pib-05.txt>.

[REQ-PART] T. Anderson, C. Wang, J. Buerkle, "Requirements for the Dynamic Partitioning of Network Elements", work in progress, August 2001, <draft-ietf-gsmp-dyn-part-reqs-00.txt>.

[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.

[RFC2212] S. Shenker, J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", RFC2215, September 1997.

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

10. Authors' Addresses

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

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
rdantu@netrake.com
214 291 1111

Avri Doria
Phone: +1 401 663 5024

Email: avri@acm.org

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

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

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