Network Working Group
Request for Comments: 3654
Category: Informational
H. Khosravi, Ed.
T. Anderson, Ed.
Intel
November 2003

Requirements for Separation of IP Control and Forwarding

Status of this Memo

Copyright Notice

Abstract

This document introduces the Forwarding and Control Element Separation (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.

Table of Contents

1. 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.

2. Definitions

Addressable Entity (AE) - A physical device that is directly addressable given some interconnect technology. For example, 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.

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

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. FEs may happen to be a single blade(or PFE), a partition of a PFE or multiple PFEs.

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 to use. However, this pre-association phase protocol is currently out of scope. 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. Again, this pre-association phase protocol is currently out of scope. 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) what is used within the ForCES protocol (see Section 6, requirement #1). However, the two capability discovery mechanisms may utilize the same FE model (see Section 5). 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. 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. These components are out of scope of the architecture and requirements for the ForCES protocol, which only involves CEs and FEs.

```
      --------------------------------
      | NE                           |
      |        -------------         |
      |        |    CE     |         |
      |        -------------         |
      |          /        \          |
      |         /          \         |
      |        /            \        |
      |       /              \       |
      |  -----------     -----------  |
      |  |   FE    |     |   FE    |  |
      |  -----------     -----------  |
      |     | | | |         | | | |   |
      |     | | | |         | | | |   |
      |     | | | |         | | | |   |
      |     | | | |         | | | |   |
      --------------------------------
            | | | |         | | | |
            | | | |         | | | |
```

4. 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 6 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]. IPv4 Forwarding functions such IP header validation, performing longest prefix match algorithm, TTL decrement, Checksum calculation, generation of ICMP error messages, etc defined in RFC 1812 should be explained.

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) 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. However, coordination between CEs is out of scope of ForCES.

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.

5. 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 6, 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.

5.1. 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, and shaping. Section 5.5 defines the minimal set of logical functions that the FE Model MUST support.

5.2. 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, and Multihomed NAT [RFC2663]. The model must be flexible enough to allow such variations in functions.

5.3. 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, and 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).

5.4. 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.

5.5. 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, the FE Model 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. This Does NOT mean off-loading of any piece of code to an FE, just that the FE Model should be able to express existing Off-loaded functions on an FE.

9) IPFLOW/PSAMP Functions
Several applications such as, Usage-based Accounting, Traffic engineering, require flow-based IP traffic measurements from Network Elements. [IPFLOW] defines architecture for IP traffic flow monitoring, measuring and exporting. The FE model SHOULD be able to express metering functions and flow accounting needed for exporting IP traffic flow information. Similarly to support measurement-based applications, [PSAMP] describes a framework to define a standard set of capabilities for network elements to sample subsets of packets by statistical and other methods. The FE model SHOULD be able to express statistical packet filtering functions and packet information needed for supporting packet sampling applications.

6. 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 5. 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 integrity of all ForCES protocol messages and protect against man-in-the-middle attacks.
   b) FE configuration information may also contain information derived from business relationships (e.g., service level agreements). Because of the confidential nature of the information, it MUST be possible to secure (make private) all ForCES protocol messages.
   c) In order to ensure that authorized CEs and FEs are participating in a NE and defend against CE or FE impersonation attacks, the ForCES architecture MUST select a means of authentication for CEs and FEs.
   d) In some deployments ForCES is expected to be deployed between CEs and FEs connected to each other inside a box over a backplane, where physical security of the box ensures that man-in-the-middle, snooping, and impersonation attacks are not possible. In such scenarios the ForCES architecture MAY rely on the physical security of the box to defend against these attacks and protocol mechanisms May be turned off.
   e) In the case when CEs and FEs are connected over a network, security mechanisms MUST be specified or selected that protect the ForCES protocol against such attacks. Any security solution used for ForCES MUST specify how it deals with such attacks.

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 L3 routing 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
   a) The ForCES protocol will be used to transport information that requires varying levels of reliability. By strict or robust reliability in this requirement we mean, no losses, no corruption, no re-ordering of information being transported and delivery in a timely fashion.
   b) Some information or payloads, such as redirected packets or packet sampling, may not require robust reliability (can tolerate some degree of losses). For information of this sort, ForCES MUST NOT be restricted to strict reliability.
   c) Payloads such as configuration information, e.g., ACLs, FIB entries, or FE capability information (described in section 6, (1)) are mission critical and must be delivered in a robust reliable fashion. Thus, for information of this sort, ForCES MUST either provide built-in protocol mechanisms or use a reliable transport protocol for achieving robust/strict reliability.
   d) Some information or payloads, such as heartbeat packets that may be used to detect loss of association between CE and FEs (see section 6, (8)), may prefer timeliness over reliable delivery. For information of this sort, ForCES MUST NOT be restricted to strict reliability.
   e) When ForCES is carried over multi-hop IP networks, it is a requirement that ForCES MUST use a [RFC2914]-compliant transport protocol.
   f) In cases where ForCES is not running over an IP network such as an Ethernet or cell fabric between CE and FE, then reliability still MUST be provided when carrying critical information of the types specified in (c) above, either by the underlying link/network/transport layers or by built-in protocol mechanisms.

7) Interconnect Independence
The ForCES protocol MUST support a variety of interconnect technologies. (refer to section 4, 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 4, requirement #7)

9) Packet Redirection/Mirroring
   a) The ForCES protocol MUST define a way to redirect packets from the FE to the CE and vice-versa. Packet redirection terminates any further processing of the redirected packet at the FE.
   b) The ForCES protocol MUST define a way to mirror packets from the FE to the CE. Mirroring allows the packet duplicated by the FE at the mirroring point to be sent to the CE while the original packet continues to be processed by the FE.

Examples of packets that may be redirected or mirrored include control packets (such as RIP, OSPF messages) addressed to the interfaces or any other relevant packets (such as those with Router Alert Option set). The ForCES protocol MUST also define a way for the CE to configure the behavior of a) and b) (above), to specify which packets are affected by each.

10) Topology Exchange
The ForCES protocol or information carried in the ForCES protocol MUST allow those FEs which have inter-FE topology information to provide that information to the CE(s).

11) Dynamic Association
The ForCES protocol MUST allow CEs and FEs to join and leave a NE dynamically. (refer to section 4, 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 4, 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.

15) Protection against Denial of Service Attacks (based on CPU overload or queue overflow)
Systems utilizing the ForCES protocol can be attacked using denial of service attacks based on CPU overload or queue overflow. The ForCES protocol could be exploited by such attacks to cause the CE to become unable to control the FE or appropriately communicate with other routers and systems. The ForCES protocol MUST therefore provide mechanisms for controlling FE capabilities that can be used to protect against such attacks. FE capabilities that MUST be manipulated via ForCES include the ability to install classifiers and filters to detect and drop attack packets, as well as to be able to install rate limiters that limit the rate of packets which appear to be valid but may be part of an attack (e.g., bogus BGP packets).

7. References

7.1. Normative References

[RFC3290] Bernet, Y., Blake, S., Grossman, D. and A. Smith, "An Informal Management Model for DiffServ Routers", RFC 3290, May 2002.

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[RFC2211] Wroclawski, J., "Specification of the Controlled-Load Network Element Service", RFC 2211, September 1997.

[RFC2212] Shenker, S., Partridge, C. and R. Guerin, "Specification of Guaranteed Quality of Service", RFC 2212, September 1997.

[RFC2215] Shenker, S. and J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", RFC 2215, September 1997.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z. and W. Weisss, "An Architecture for Differentiated Service", RFC 2475, December 1998.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 14, RFC 2914, September 2000.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

7.2. Informative References

[RFC3532] Anderson, T. and J. Buerkle, "Requirements for the Dynamic Partitioning of Switching Elements", RFC 3532, May 2003.

[IPFLOW] Quittek, et al., "Requirements for IP Flow Information Export", Work in Progress, February 2003.

[PSAMP] Duffield, et al., "A Framework for Passive Packet Measurement ", Work in Progress, March 2003.

8. Security Considerations

See architecture requirement #5 and protocol requirement #2.

9. 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
Department of Computer Science
University of North Texas,
Denton, Texas, 76203

Phone: 940 565 2822
EMail: rdantu@unt.edu

Avri Doria
ETRI
161 Gajeong-dong, Yuseong-gu
Deajeon 305-350 Korea

EMail: avri@acm.org

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
Nokia Research Center
5 Wayside Road
Burlington, MA 01803
Phone: +1 781 993 3858
EMail: margaret.wasserman@nokia.com

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

10. 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

11. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.