The Internet Protocol Journal

July 2021 Volume 24, Number 2

A Quarterly Technical Publication for Internet and Intranet Professionals

In This Issue

You can download IPJ back issues and find subscription information at:
www.protocoljournal.org

ISSN 1944-1134

From The Editor

Since the launch of *The Internet Protocol Journal* in 1998, we have covered several aspects of the core technologies used in the global Internet and in enterprise networks. Routing protocols such as the *Border Gateway Protocol* (BGP) continue to play a critical role in the operation of these networks, but other routing protocols are being developed by the *Internet Engineering Task Force* (IETF) for use in data center environments. One such protocol is the *Routing in Fat Trees Protocol* (RIFT).

Route *aggregation* is used to summarize a set of specific routing-table entries into a single, less-specific route, in order to reduce the size of routing tables. Disaggregation is the opposite of aggregation whereby an aggregate route is divided into several more-specific routes. In our first article, Bruno Rijsman explains how automatic disaggregation is accomplished in RIFT.

Security has also been a recurring theme in this journal. Most of the protocols used in today's Internet were originally designed without comprehensive security in mind, but the IETF has produced security enhancements for many of the core protocols. Securing the routing system itself has proven challenging because it requires widespread deployment in order to be effective. Starting with our next issue, Geoff Huston presents a two-part article entitled "A Survey on Securing Inter-Domain Routing." Make sure your subscription details are up to date!

The IETF is, of course, not the only organization that produces standards for computer networks. Our second article, by William Stallings, is an overview of *Network Functions Virtualization*, an emerging set of standards being developed by the *European Telecommunications Standards Institute* (ETSI).

The generous support of individuals and organizations makes publication of this journal possible. We are pleased to welcome our latest sponsor, *The APNIC Foundation*. More information about the foundation is available on page 30 of this issue.

—Ole J. Jacobsen, Editor and Publisher
ole@protocoljournal.org

Automatic Disaggregation in the Routing in Fat Trees Protocol

by Bruno Rijsman

Routing in Fat Trees (RIFT) is a new routing protocol being defined in the *Internet Engineering Task Force* (IETF).[1] RIFT is optimized for large networks that have a highly structured topology such as fat tree, *Clos*, or similar topologies. It is typically used as a scalable and fast-converging *Interior Gateway Protocol* (IGP) for the underlay in data centers, but it has other use cases as well.[2]

RIFT brings several innovations to the table without requiring any changes to existing networking hardware ("silicon"), including:

- *Zero Touch Provisioning* (ZTP) virtually eliminates the need for configuration and auto-detects miscabling.
- RIFT is *anisotropic:* it is a link-state protocol north-bound and a distance-vector protocol south-bound, combining the advantages of link-state with the advantages of distance-vector/path-vector.
- RIFT is inherently loop-free, allowing it to distribute traffic across all available paths (not just the shortest path).
- A built-in flooding-reduction mechanism greatly reduces flooding traffic in densely connected topologies such as fat trees.
- With the automatic aggregation feature, in the absence of failures, each node needs only a single multi-path default route pointing north. This feature reduces the size of the routing tables at or close to the leaf nodes, and hence the cost of top-of-rack switches.
- With the automatic disaggregation feature, if a failure occurs the north-bound default route is automatically disaggregated into more specific routes, but only to the extent needed to route around the failure.
- RIFT supports large data-center networks without the need for splitting the network into multiple areas.
- RIFT offers very fast convergence—even in very large networks.
- Because of the simplicity of RIFT functionality on leaf switches, you can easily run it on servers; this feature is also known as *Routing on The Host* (RoTH). It enables support for multi-homed servers with automatic recovery from link and node failures.
- Model-based (Thrift) specification of the routing protocol messages accelerates development, enhances interoperability, and most importantly, improves security by removing most message-parsing vulnerabilities.

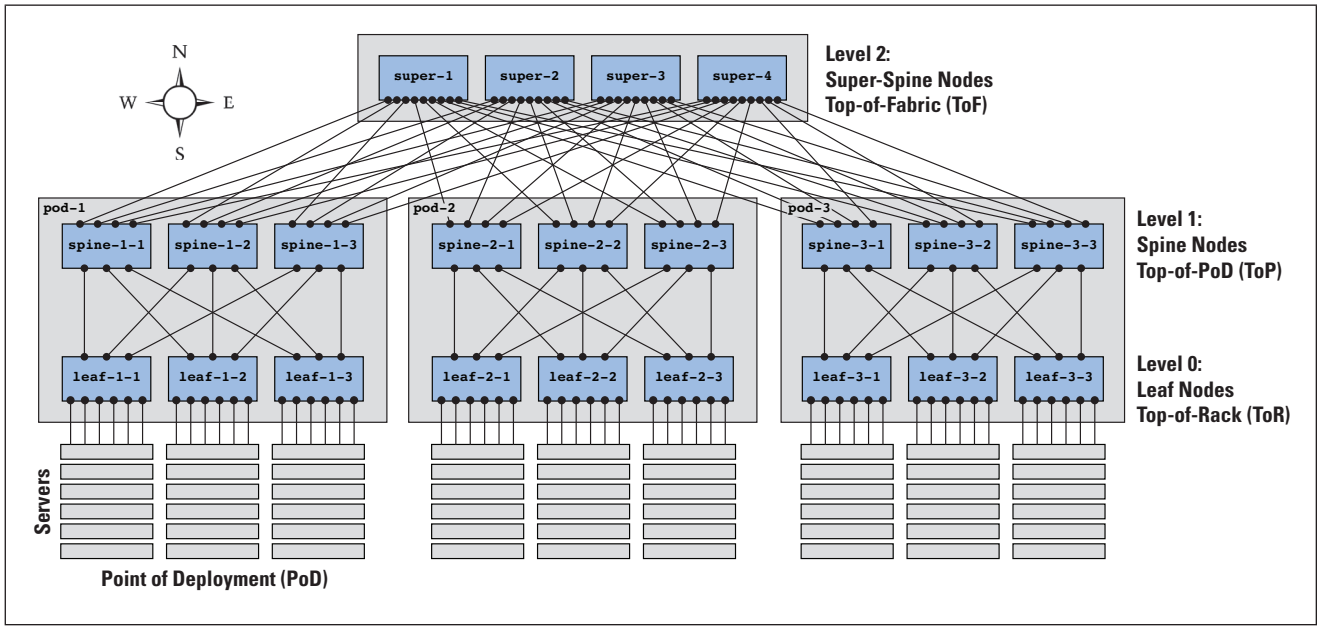
RIFT offers an open-source implementation[3] and at least one commercial implementation.[4]

In this article, we focus on one feature of RIFT: automatic aggregation and disaggregation, which is one of the most novel and most interesting innovations in the RIFT protocol. For a more general overview of RIFT, see the presentation at APNIC[5] or the recently released (and free) *Day One book on RIFT*.[6] For a discussion of link-state routing in data centers (including RIFT), see the article "Recent Developments in Link State on Data-Center Fabrics," also published in this journal.[7]

Introduction to RIFT

You can use RIFT in topologies where it makes sense to speak of north and south directions, which allows you to divide the nodes into levels, including fat-tree data center topologies such as the one shown in Figure 1.

Figure 1: Fat-Tree Data Center Topology



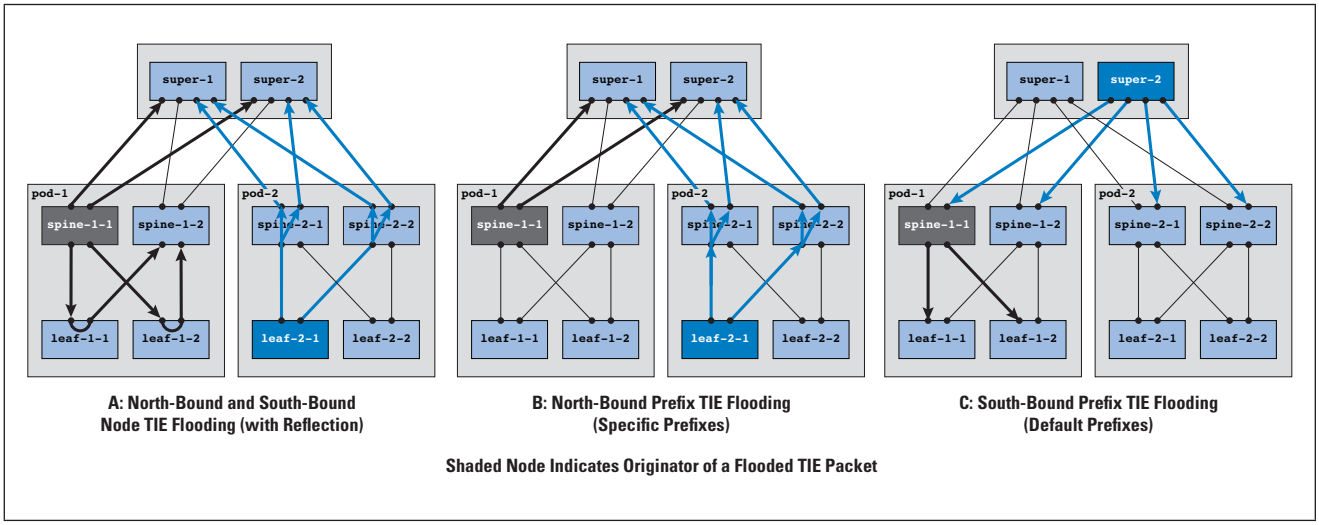
In many respects, RIFT is a link-state protocol similar to *Intermediate System-to-Intermediate System* (IS-IS):

- RIFT routers exchange hello packets, called *Link Information Element* (LIE) packets, to establish adjacencies with neighbor routers.
- RIFT routers originate link-state packets, called *Topology Information Element* (TIE) packets, to describe the state, adjacencies, originated prefixes, disaggregated prefixes, and other information about the router.
- RIFT reliably floods the link-state packets across the network. It uses *Topology Information Description Element* (TIDE) packets to summarize the contents of the link-state database and *Topology Information Request Element* (TIRE) packets to acknowledge and request TIE packets. Together, TIDEs and TIREs are used to make the flooding reliable.

- RIFT stores link-state packets in its *Link State Database* (LSDB).
- RIFT runs the *Shortest Path First* (SPF) algorithm on the topology stored in the link-state database to compute the shortest path to each destination.

RIFT is unique in that it has different rules for flooding TIE packets in both the north-bound south-bound directions, as shown in Figure 2:

Figure 2: TIE Flooding Rules



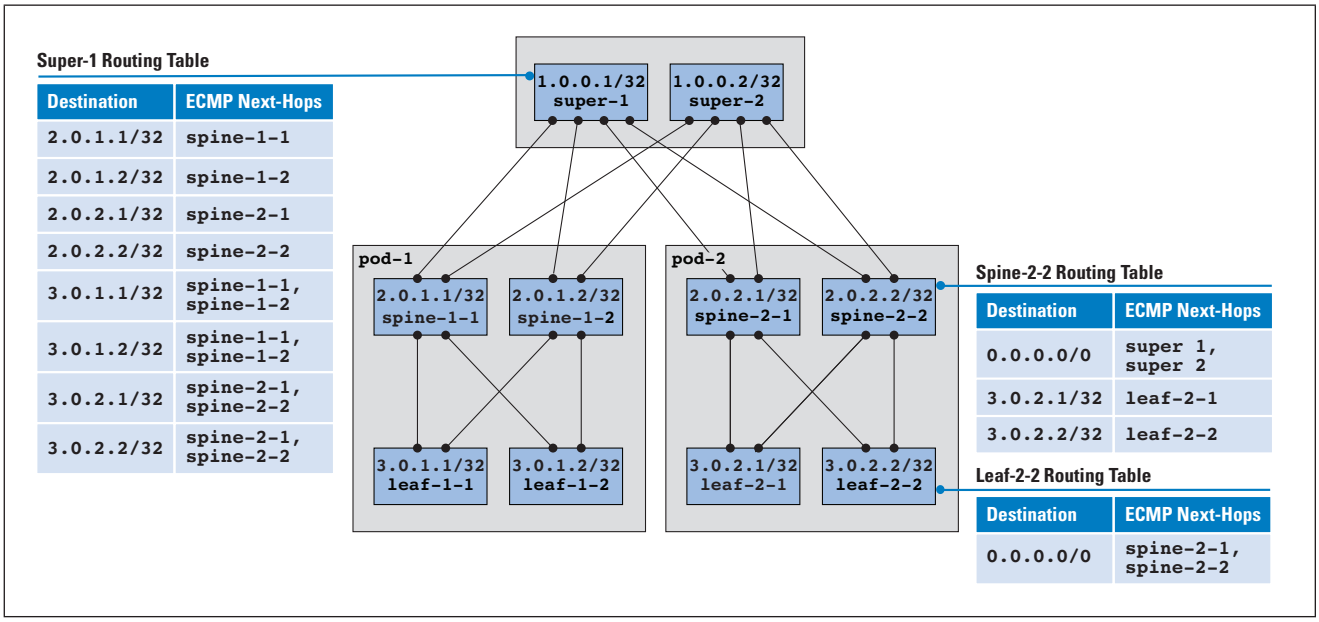
- Each node advertises its adjacencies in node TIEs that are flooded in both the north-bound and south-bound directions. Furthermore, the level just below the originator of the node TIE "reflects" the node TIEs back-up. This reflection allows nodes to discover the adjacencies of other nodes at the same level (refer to diagram A in Figure 2).
- Each node advertises its local prefixes in prefix TIEs, which are flooded only in the north-bound direction. (Shown in diagram B in Figure 2).
- Each node advertises a fabric default route (typically `0.0.0.0/0` and `::/0`) in prefix TIEs that are flooded exactly one hop (but no further) in the south-bound direction (see diagram C in Figure 2). The top-of-fabric nodes always originate a default, and the lower nodes originate a default only if they have received at least one default from a parent. This model makes the south-bound flooding similar to distance-vector routing and it is the reason that RIFT is colloquially described as link-state towards the spine and distance-vector towards the leaves.
- RIFT also allows for east-west "short-cut" links and has flooding rules for those links (not shown in the figure).

- RIFT also includes a flooding-reduction mechanism that avoids multiple copies of the same TIE being sent to the same node (that mechanism is not shown here). For example, in diagrams A and B in Figure 2 node super-1 receives two identical copies of the TIE from leaf-2-1.

After the TIEs are flooded across the network in the manner described previously, the RIFT nodes run the SPF algorithm to compute the routing tables. Actually, RIFT does at least two SPF runs: one for the north-bound and one for the south-bound direction.

Figure 3 shows an example of typical RIFT routing tables (in the absence of failures):

Figure 3: Typical RIFT Routing Tables in the Absence of Failures



We can see the following routes:

- *Specific routes for all south-bound traffic:* These routes are typically host /32 (for IPv4) or /128 (for IPv6) routes.
- *Fabric default routes for all north-bound traffic:* These routes are typically `0.0.0.0/0` or `::/0` routes.

Both the north-bound default routes and the south-bound, specific routes, are multi-path routes, distributing the traffic across all available paths. The next-hops can be weighted according to the bandwidth available on each path.

RIFT Automatic Aggregation and Disaggregation

The *aggregation*[8] concept has existed in routing protocols since the beginning. Aggregation allows you to summarize a set of specific routes by a single, less-specific route, called the *aggregate route*. The most common use case for aggregation is to reduce the size of the routing table by summarizing unneeded details.

The concept of *disaggregation*[9] has also been used for a long time. Disaggregation is the opposite of aggregation: it takes a single less-specific route (the aggregate route) and divides it into several more-specific routes. The most common use case for disaggregation is traffic engineering.

In existing protocols such as the *Border Gateway Protocol* (BGP), *Open Shortest Path First* (OSPF), and IS-IS, aggregation and disaggregation typically are manually configured for optimization purposes. In RIFT, on the other hand, aggregation and disaggregation are automatic and always enabled. RIFT automatically aggregates routes (typically to the default route) wherever possible. And RIFT automatically disaggregates routes wherever needed, for example, because of link or node failures.

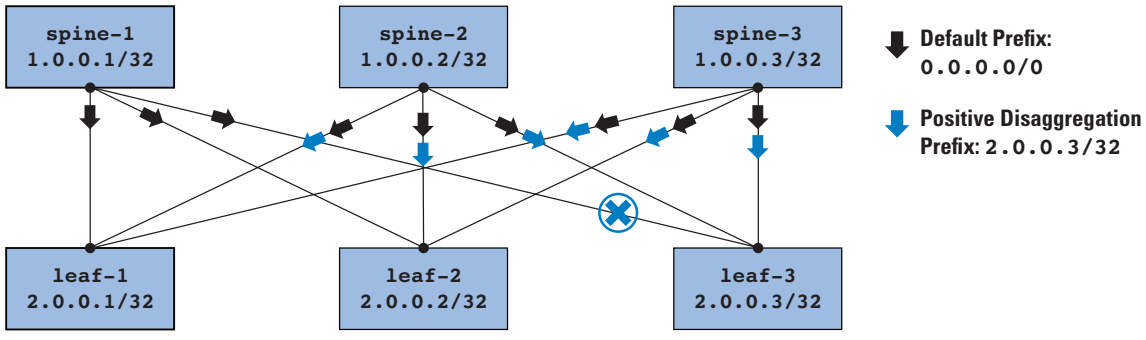
Disaggregation actually has two modes in RIFT:

- *Positive Disaggregation* works by advertising a more specific route to "attract" traffic to a repair path away from a failed path. Advertising more-specific prefixes is exactly how disaggregation works in existing protocols.
- *Negative Disaggregation* works by advertising a so-called *negative prefix* to "repel" traffic away from a failed path towards a repair path. This new mechanism does not have an equivalent in existing widely deployed protocols. Negative disaggregation is needed only in certain large topologies, namely multi-plane topologies.

RIFT Positive Disaggregation

Earlier we saw that RIFT normally uses default routes for northbound traffic, which reduces the size of the forwarding tables, but it may cause traffic to be black-holed when a link failure occurs (refer to Figure 4):

Figure 4: Positive Disaggregation in a Two-Level Fabric



Leaf-1 Routing Information Base (RIB)

Destination	ECMP Next-Hops
0.0.0.0/0	spine-1, spine-2, spine-3
2.0.0.3/32	spine-2, spine-3

Leaf-1 Forwarding Information Base (FIB)

Destination	ECMP Next-Hops
0.0.0.0/0	spine-1, spine-2, spine-3
2.0.0.3/32	spine-2, spine-3

Consider a link failure between spine-1 and leaf-3, as shown in Figure 4. When leaf-1 wants to send traffic to leaf-3 and follows its north-bound *Equal-Cost Multi-path Routing* (ECMP) default route, it might select spine-1 as the next hop, which black-holes the traffic.

To avoid such black-holing of traffic, spine-2 and spine-3 each automatically triggers positive disaggregation. Following is the sequence of events from the perspective of spine-2, but the same sequence of events happens at spine-3:

1. Spine-2 has a south-bound route `2.0.0.3/32`, whose next-hop is leaf-3.
2. Spine-2 knows the adjacencies of spine-1 because the leaves reflected the spine-1 node TIE of spine-1 to spine-2. Hence, spine-2 knows that spine-1 does not have an adjacency with leaf-3 and that spine-1 cannot reach `2.0.0.3/32`.
3. Spine-2 automatically initiates positive disaggregation by flooding a positive disaggregation prefix TIE containing prefix `2.0.0.3/32` in the south-bound direction (the blue arrows in Figure 4).
4. Leaf-1 and leaf-2 install the more-specific route prefix `2.0.0.3/32` in their forwarding table. In the end, this route ends up being a two-way ECMP route across spine-2 and spine-3 (but not spine-1). Note that it takes a finite amount of time for the route to reach its full ECMP next-hop set, which may cause transitory in-cast issues (these issues can be addressed with implementation-specific mechanisms).
5. Leaf-1 and leaf-2 still rely on the default route for all other destinations. This route is a three-way ECMP route across all three spines.

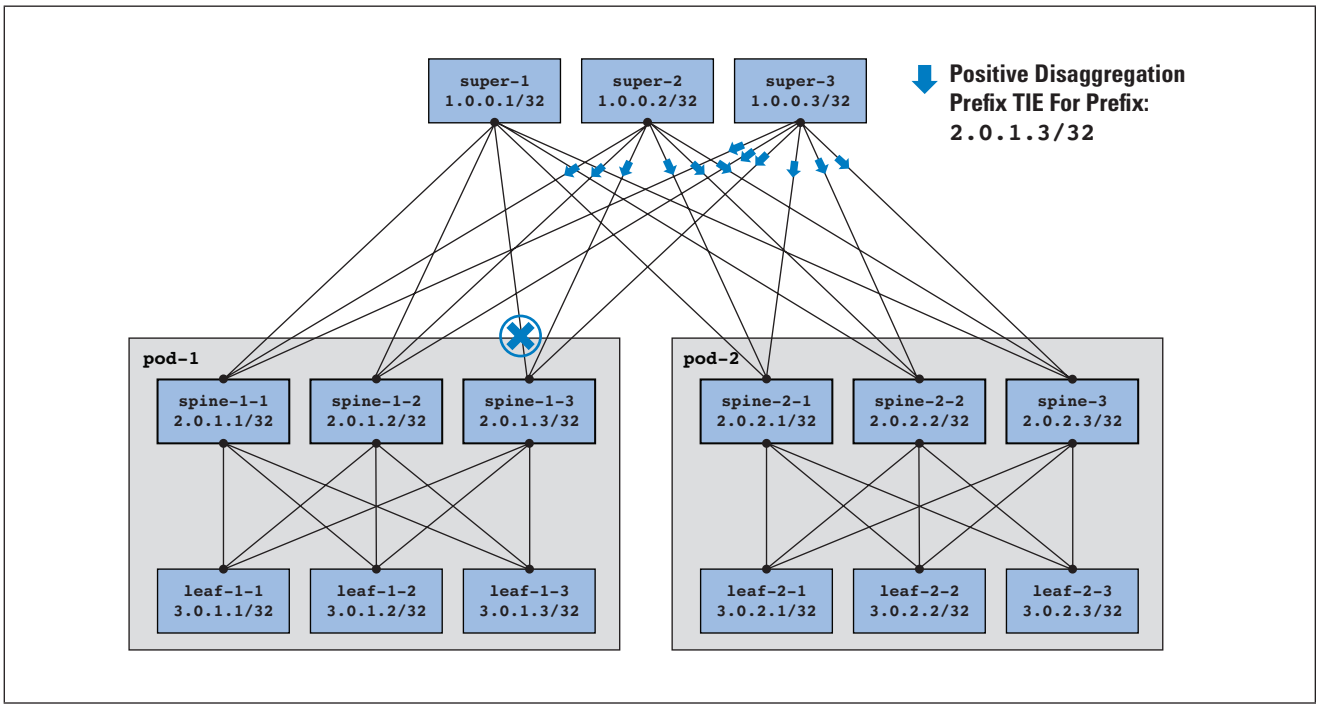
In summary, spine-2 and spine-3 detected that a link was broken in the topology, and they automatically initiated positive disaggregation to "attract" the traffic away from the failed path (spine-1) towards a repair path (themselves).

We now consider positive disaggregation in a more-complex scenario, namely a three-level fabric. In Figure 5, the link from super-1 to spine-1-3 has failed.

Super-2 and super-3 will automatically initiate positive disaggregation for prefix `2.0.1.3/32` because:

1. Super-2 and super-3 have a south-bound route for prefix `2.0.1.3/32` with only one next-hop, namely spine-1-3.
2. Super-2 and super-3 know that super-1 does not have an adjacency with spine-1-3.
3. Super-2 and super-3 conclude that super-1 can no longer reach `2.0.1.3/32`. Hence, they initiate positive disaggregation for that prefix.

Figure 5: Positive Disaggregation Repairs a Single Failure in a Three-level Fabric



However, at this point super-2 and super-3 will not yet initiate positive disaggregation for prefixes `3.0.1.1/32`, `3.0.1.2/32`, and `3.0.1.3/32` because:

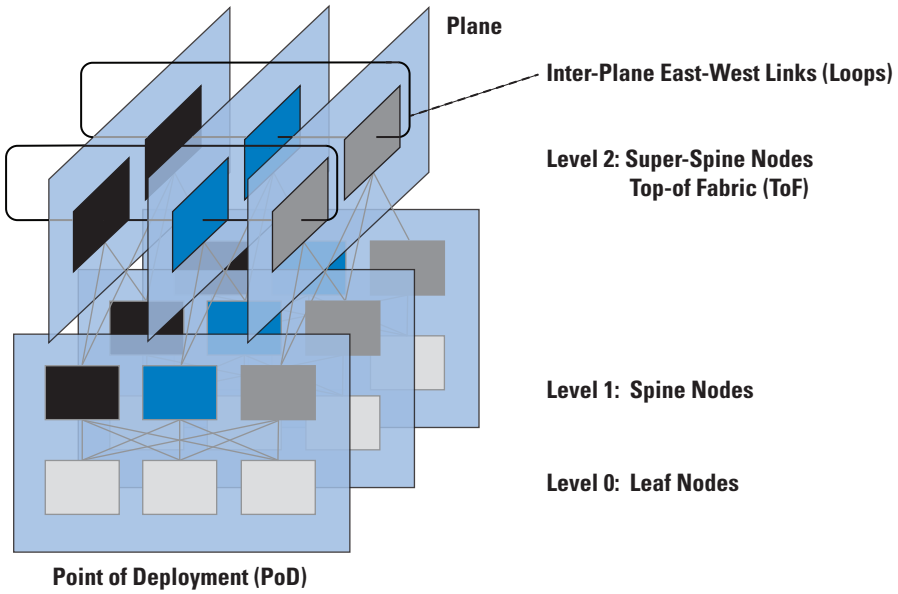
1. Super-2 and super-3 have routes for prefixes `3.0.1.1/32`, `3.0.1.2/32`. and `3.0.1.3/32`, each with three ECMP next-hops, namely spine-1-1, spine-1-2, and spine-1-3.
2. Super-2 and super-3 know that super-1 does not have an adjacency with spine-1-3, but it does still have an adjacency with spine-1-1 and spine-1-2.
3. Super-2 and super-3 conclude that although super-1 can still reach `3.0.1.1/32`, `3.0.1.2/32`, and `3.0.1.3/32:` not through spine-1-3 but still through spine-1-1 and spine-1-2. Hence, they do not initiate positive disaggregation for those prefixes.

We leave it as an exercise for the reader to verify that super-2 and super-3 will initiate disaggregation for `3.0.1.1/32`, `3.0.1.2/32`, and `3.0.1.3/32` when all links from super-1 to pod-1 are broken.

Multi-Plane Topologies

In the fat-tree topologies that we have considered thus far, every spine is connected to every super-spine. When the network becomes large, you reach a point where the super-spines don't have enough ports to connect to every spine. Such networks often use a multi-plane topology such as the one shown in Figure 6 on the following page.

Figure 6: A Multi-Plane Topology (with East-West Inter-Plane Links)



For now, ignore the loops that connect the super-spines together; they are explained later. In a multi-plane topology, the spines and super-spines are partitioned into planes. In Figure 6, we have a blue plane, a black plane, and a dark grey plane. The super-spines in a plane are connected only to the spines in that same plane.

In such a multi-plane topology, the RIFT positive disaggregation mechanism does not always work because node TIEs are reflected only between super-spines in in the same plane. The dark grey super-spines, for example, do not know the adjacencies of the blue or black super-spines. Hence, a super-spine in one plane cannot detect that a super-spine in a different plane has lost connectivity to some set of prefixes.

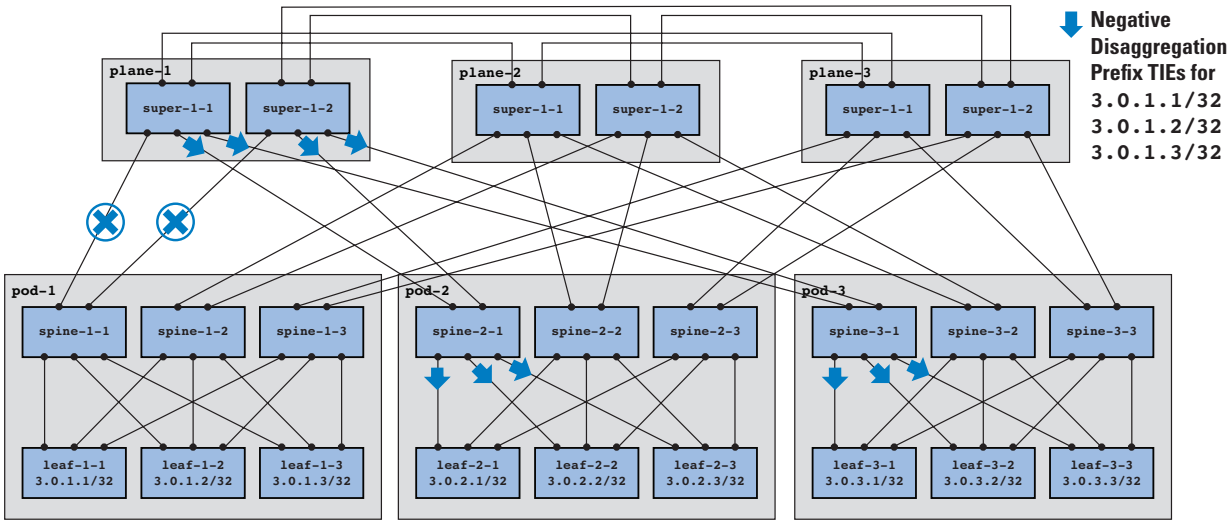
Negative Disaggregation

RIFT uses a different disaggregation mechanism, called *negative disaggregation*, to recover from failures in multi-plane topologies.

To make negative disaggregation work, the super-spines in different planes need to be interconnected using east-west inter-plane links as shown in Figures 6 and 7. These east-west inter-plane links are used only for control-plane traffic, and they do not carry user traffic (so they can be low-bandwidth links).

To understand negative disaggregation, consider the following multi-plane topology:

Figure 7: Negative Disaggregation



Leaf-2-2 Routing Information Base (RIB)

Destination	ECMP Next-Hops
0.0.0.0/0	spine-2-1, spine-2-2, spine-2-3
3.0.1.1/32	negative spine 2-1
3.0.1.2.32	negative spine 2-1
3.0.1.3/32	negative spine 2-1

Leaf-2-2 Forwarding Information Base (FIB)

Destination	ECMP Next-Hops
0.0.0.0/0	spine-2-1, spine-2-2, spine-2-3
3.0.1.1/32	spine 2-2, spine 2-3
3.0.1.2.32	spine 2-2, spine 2-3
3.0.1.3/32	spine 2-2, spine 2-3

Triggering Negative Disaggregation

The super-spines run the south-bound *Shortest Path First* (SPF) algorithm twice:

1. The *normal SPF run* excludes the east-west inter-plane links. The resulting routes are installed in the *Routing Information Base* (RIB) and the *Forwarding Information Base* (FIB).

2. The *special SPF run* includes the east-west inter-plane links. The resulting routes are not installed in the RIB or FIB. If the special SPF run finds any extra reachable prefixes that were not reachable in the normal SPF run, then those extra prefixes are declared to be "fallen leaves," and they trigger negative disaggregation.

In Figure 7, from the perspective of super-1-1 and super-1-2, the prefixes in pod-1 are fallen leaves because they can be reached only through other planes (that is, using east-west inter-plane links). The special SPF run will find the prefixes in pod-1, but the normal SPF run will not find the prefixes in pod-1.

After a super-spine detects fallen-leaf prefixes, it advertises those prefixes in a negative disaggregation prefix TIE, which is flooded south in the topology.

The super-spine is telling the rest of the network "don't send any traffic destined to the fallen-leaf prefix to me because I cannot reach it."

In a sense, negative disaggregation is the opposite of positive disaggregation. In positive disaggregation, the repair path advertises a positive disaggregation route to *attract* the traffic away from the broken path. In negative disaggregation, the broken bath advertises a negative disaggregation prefix to *repel* traffic away towards the repair path. The mechanism for choosing the repair path is described in the sections that explain negative next-hop-to-positive next-hop translation.

Propagation of Negative Disaggregation

Unlike positive disaggregation (which is never propagated), negative disaggregation can be recursively propagated southwards. RIFT uses special rules for south-bound flooding of negative disaggregation prefix TIEs: a node propagates a negative disaggregation prefix *only* if it was received from *all* of the parent nodes, meaning that this node does not have any path left to the fallen leaf.

In Figure 7, spine-2-1 has received a negative disaggregation prefix TIE for the prefixes in pod-1 from both of its parent nodes, namely super-1-1 and super-1-2. Hence, spine-2-1 propagates the negative disaggregation prefix TIE further south-bound. The same happens at spine-3-1.

Negative Disaggregation in the RIB

When a node receives a negative disaggregation prefix TIE, it is stored in the LSDB and it takes part in the SPF calculation, just like a normal prefix TIE. However, the resulting route is installed in the RIB using a negative next-hop instead of a positive next-hop.

In Figure 7, leaf-2-2 has a north-bound default route `0.0.0.0/0` with three ECMP next-hops: spine-2-1, spine-2-2, and spine-2-3. These next-hops are normal (that is, positive) next-hops; the traffic will be distributed across all three spines in the pod.

Leaf-2-2 also has north-bound more-specific routes `3.0.1.x/32` (the prefixes in pod-1) with a negative next-hop spine-2-1. A negative next-hop in the RIB is a control-plane construct, meaning "don't send the traffic to this next-hop." The intent of this negative next-hop is to avoid sending traffic for `3.0.1.x/32` into plane-1 because plane-1 is disconnected from pod-1.

Note that a negative next-hop is something different from a discard next-hop. A discard next-hop causes traffic to be dropped. A negative next-hop causes traffic to be sent somewhere else using a less-specific route. We will now explain how it works.

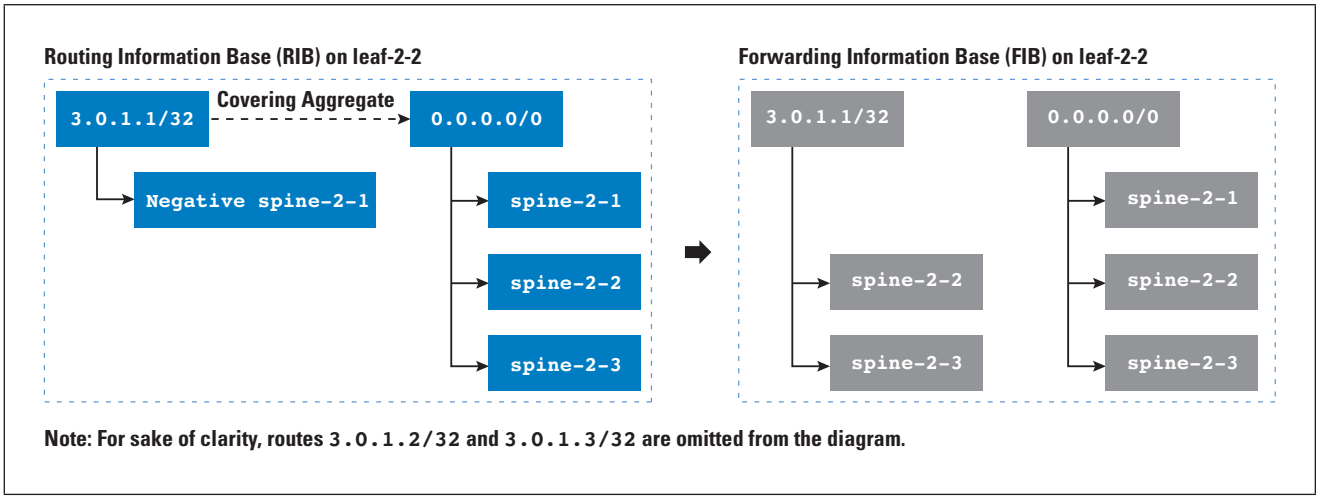
Negative Disaggregation in the FIB

Negative next-hops do not exist in current-generation forwarding hardware; they are a RIFT abstraction that exists only in the control plane and not in the forwarding plane.

When a RIFT route is installed from the RIB into the FIB, the RIB negative next-hop (where not to send the traffic) is translated into positive next-hops (where to send the traffic to instead).

Figure 8 illustrates how this translation works:

Figure 8: Translating Negative Next-Hops in the RIB into Positive Next-Hops in the FIB



What is happening in this simple example is the following:

- We have a route to `3.0.1.1/32`, which has a negative next-hop.
- We find the most specific aggregate route that covers this route, which is the default route, `0.0.0.0/0` in this case.
- We add up the next-hops of routes `3.0.1.1/32` and `0.0.0.0/0`, keeping in mind that a negative and positive next-hop cancel each other out.

We started with a route for `3.0.1.1/32` with negative next-hop spine-2-1. We translated the negative next-hop spine-2-1 into the complementary positive ECMP next-hops spine-2-2 and spine-2-3. These translated next-hops are stored in the FIB.

Further Reading

For more-detailed information about RIFT disaggregation, see Pascal Thubert's slides on negative disaggregation presented at the IETF [10], the RIFT-Python open-source documentation,[11,12] or my blog post on the topic,[13] which goes into more detail.

Conclusion

In this article, we have introduced the RIFT protocol and described how RIFT uses automatic aggregation (north-bound default routes) to reduce the size of the routing table.

We have explained how RIFT uses automatic disaggregation to reroute traffic around failed links and nodes. We have further described the two types of disaggregation in RIFT, namely *Positive Disaggregation* and *Negative Disaggregation*.

Positive Disaggregation *attracts* traffic to the repair path by advertising more-specific routes, except that RIFT advertises these routes automatically instead of as a result of manual configuration. Negative disaggregation *repels* traffic away from the broken path. This approach is novel in that it relies on the new concept of a "negative next-hop." These negative next-hops are translated into normal positive next-hops in the data-plane hardware.

Acknowledgements

I would like to thank Tony Przygienda (the father of RIFT), Pascal Thubert (the father of negative disaggregation), and Melchior Aelmans (one of my co-authors on the RIFT book) for our frequent and deep conversations on the RIFT protocol, and for their review of this article. I would like to thank Mariano Scazzariello and Tommaso Caiazzi (both PhD students at Roma University in Italy) for implementing negative disaggregation in RIFT-Python and their extensive testing of RIFT at scale.[16]

References

[1] Tony Przygienda, Alankar Sharma, Pascal Thubert, Bruno Rijsman, and Dmitry Afanasiev, "RIFT: Routing in Fat Trees," Internet Draft, Work in Progress, **draft-ietf-rift-rift-12**, May 2020

[2] Yuehua Wei, Zheng Zhang, Dmitry Afanasiev, Tom Verhaeg, Jaroslaw Kowalczyk, and Pascal Thubert, "RIFT Applicability," Internet Draft, Work in Progress, **draft-ietf-rift-applicability-03**, October 2020.

[3] RIFT-Python, an open-source implementation of RIFT in Python, Bruno Rijsman and other contributors: **https://github.com/brunorijsman/rift-python**

[4] Juniper Networks, "RIFT User Guide for Junos OS," **https://www.juniper.net/documentation/en_US/junos/information-products/pathway-pages/config-guide-routing/config-guide-routing-rift.html**

[5] Antoni Przygienda and Zhaohui (Jeffrey) Zhang, "Routing in Fat Trees; A New DC Routing Protocol," **https://www.slideshare.net/apnic/routing-in-fat-trees**

[6] Melchior Aelmans, Olivier Vandezande, Bruno Rijsman, Jordan Head, Christan Graf, Leonardo Alberro, Hitesh Mali, and Oliver Steudler, *Day One: Routing In Fat Trees (RIFT), A complete look at the cutting edge protocol,* Juniper Networks Books. **https://www.juniper.net/documentation/en_US/day-one-books/DO_RIFT.pdf**

[7] Russ White and Melchior Aelmans, "Recent Developments in Link State on Data-Center Fabrics," *The Internet Protocol Journal*, Volume 22, Number 2, September 2020.

[8] Juniper Networks, "Understanding Route Aggregation," **https://www.juniper.net/documentation/en_US/junos/topics/concept/policy-aggregate-routes.html**

[9] Geoff Huston, "BGP More Specifics: Routing Vandalism or Useful?" Published on the RIPE NCC website: **https://labs.ripe.net/Members/gih/bgp-more-specifics-routing-vandalism-or-useful**

[10] Pascal Thubert, "Negative Disaggregation," **https://datatracker.ietf.org/doc/slides-103-rift-negative-disaggregation/**

[11] Bruno Rijsman, "RIFT-Python Positive Disaggregation Feature Guide," **https://github.com/brunorijsman/rift-python/blob/master/doc/positive-disaggregation-feature-guide.md**

[12] Bruno Rijsman, "RIFT-Python Negative Disaggregation Feature Guide," **https://github.com/brunorijsman/rift-python/blob/master/doc/negative-disaggregation-feature-guide.md**

[13] Bruno Rijsman Blog, "Automatic Disaggregation in the Routing in Fat Trees (RIFT) Protocol," **https://hikingandcoding.wordpress.com/2020/07/22/rift-disaggregation/**

[14] Bruno Rijsman GitHub Page: **https://github.com/brunorijsman**

[15] Wojciech Kozlowski, Stephanie Wehner, Rodney Van Meter, Bruno Rijsman, Angela Cacciapuoti, Marcello Caleffi, and Shota Nagayama, "Architectural Principles for a Quantum Internet," Internet Draft, Work in Progress, September 2020, **draft-irtf-qirg-principles-05**

[16] Tommaso Caiazzi, Mariano Scazzariello, Lorenzo Ariemma, "VFTGen: a Tool to Perform Experiments in Virtual Fat Tree Topologies," **http://dl.ifip.org/db/conf/im/im2021demo/213179.pdf**

BRUNO RIJSMAN is a software engineer and architect working mainly on networking protocols. Over the past 25 years, he has held technical and leadership roles at network equipment vendors, including Juniper Networks, Verivue, and Lucent Technologies. He currently spends most of his time on open-source projects [14], which include RIFT and quantum networking[15].
E-mail: **brunorijsman@gmail.com**

Network Functions Virtualization

by William Stallings

Network Functions Virtualization (NFV) originated from discussions among major network operators and carriers about how to improve network operations in the high-volume multimedia era. These discussions resulted in the publication of the original 2012 NFV White Paper by an NFV Industry Specification Group within the *European Telecommunications Standards Institute* (ETSI).[1] In the white paper, the group listed as the overall objective of NFV the leveraging of standard IT virtualization technology to consolidate many network equipment types onto industry-standard high-volume servers, switches, and storage, which could be located in data centers, network nodes, and at the end-user premises.

The white paper highlights that the source of the need for this new approach is that networks include a large and growing variety of proprietary hardware appliances, leading to the following negative consequences:

- New network services may require additional different types of hardware appliances and finding the space and power to accommodate these boxes is becoming increasingly difficult.
- New hardware means additional capital expenditures.
- After new types of hardware appliances are acquired, operators are faced with the rarity of skills necessary to design, integrate, and operate increasingly complex hardware-based appliances.
- Hardware-based appliances rapidly reach end of life, requiring much of the procure-design-integrate-deploy cycle to be repeated with little or no revenue benefit.
- As technology and services innovation accelerates to meet the demands of an increasingly network-centric IT environment, the need for an increasing variety of hardware platforms inhibits the introduction of new revenue-earning network services.

The NFV approach moves away from the dependence on a variety of hardware platforms to the use of a small number of standardized platform types, with virtualization techniques used to provide the needed network functions. In the white paper, the group expresses the belief that the NFV approach is applicable to any data-plane packet-processing and control-plane function in fixed and mobile network infrastructures.

NFV deployment has become increasingly widespread, being used by telecommunications providers, cloud service providers, and large enterprises, such as in the banking and financial services industry.[2] Perhaps the main driver for NFV is 5G wireless networks.[3] NFV is an integral part of 5G and is indeed required by 5G standards.[4]

Concepts

NFV builds on standard *Virtual Machine* (VM) technologies, extending their use into the networking domain. This departure from traditional approaches to the design, deployment, and management of networking services is significant. NFV decouples network functions, such as *Network Address Translation* (NAT), firewalling, intrusion detection, *Domain Name System* (DNS), and caching, from proprietary hardware appliances so they can run as software on VMs.

Virtual-machine technology enables migration of dedicated application and database servers to *Commercial Off-The-Shelf* (COTS) x86 servers. You can apply the same technology to network-based devices, including:

- *Network Function Devices:* Such as switches, routers, network access points, and deep packet inspectors
- *Network-related Compute Devices:* Such as firewalls, intrusion detection systems, and network management systems
- *Network-attached Storage:* File and database servers attached to the network

In traditional networks, all network elements are enclosed boxes, and hardware cannot be shared. Each device requires additional hardware for increased capacity, but this hardware is idle when the system is running below capacity. With NFV, however, network elements are independent applications that are flexibly deployed on a unified platform comprising standard servers, storage devices, and switches. In this way, software and hardware are decoupled, and capacity for each application is increased or decreased by adding or reducing virtual resources.

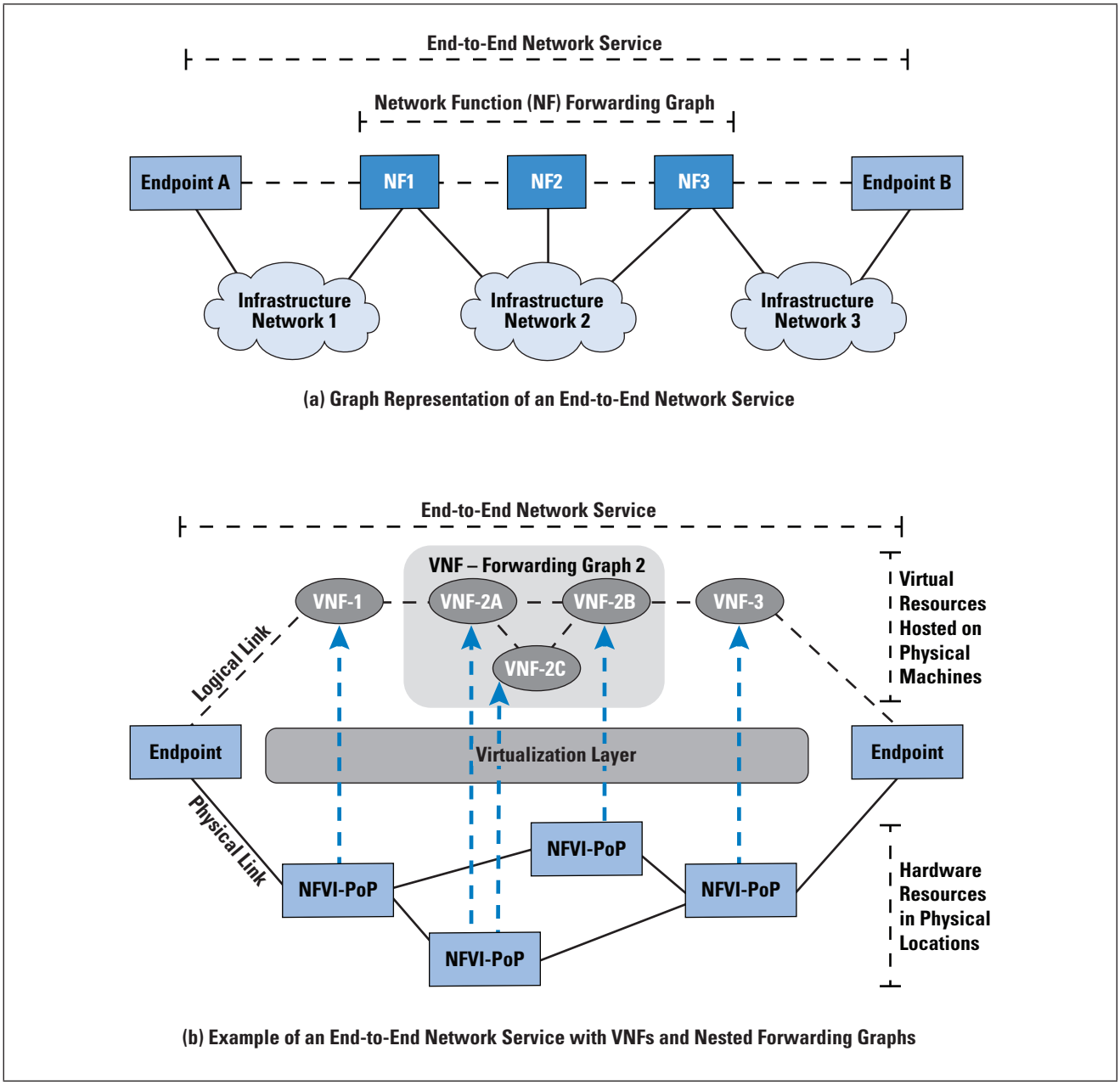
Consider a simple example from the *NFV Architectural Framework* document. Figure 1a shows a physical realization of a network service. At a top level, the network service consists of *endpoints* connected by a forwarding graph of network functional blocks, called *Network Functions* (NFs). Examples of NFs are firewalls, load balancers, and wireless network access points. In the Architectural Framework, NFs are viewed as distinct physical nodes. The endpoints are outside the scope of the NFV specifications and include all customer-owned devices. So, in the figure, endpoint A could be a smartphone and endpoint B a *Content Delivery Network* (CDN) server.

Figure 1a highlights the network functions that are relevant to the service provider and customer. The interconnections among the NFs and endpoints are depicted by dashed lines, representing logical links. These logical links are supported by physical paths through infrastructure networks (wired or wireless).

Figure 1b shows a virtualized network service configuration that could be implemented on the physical configuration of Figure 1a. *Virtual Network Function* (VNF) 1 provides network access for endpoint A, and VNF 2 provides network access for B.

The figure also depicts the case of a nested VNF forwarding graph (VNF-FG-2) constructed from other VNFs (that is, VNF-2A, VNF-2B, and VNF-2C). All of these VNFs run as virtual machines on physical machines, called *Points of Presence* (PoPs). This configuration illustrates several important points. First, VNF-FG-2 consists of three VNFs even though ultimately all of the traffic transiting VNF-FG-2 is between VNF-1 and VNF-3. The reason for this situation is that three separate and distinct network functions are being performed. For example, it may be that some traffic flows need to be subjected to a traffic policing or shaping function, which could be performed by VNF-2C. So, some flows would be routed through VNF-2C while others would bypass this network function.

Figure 1: A Simple NFV Configuration Example



A second observation is that two of the VMs in VNF-FG-2 are hosted on the same physical machine. Because the two VMs perform different functions, they need to be distinct at the virtual resource level but can be supported by the same physical machine. But this setup is not required, and a network management function may at some point decide to migrate one of the VMs to another physical machine, for reasons of performance. This movement is transparent at the virtual resource level.

Principles

As Figure 1 suggests, the VNFs are the building blocks used to create end-to-end network services. Three key NFV principles are involved in creating practical network services:

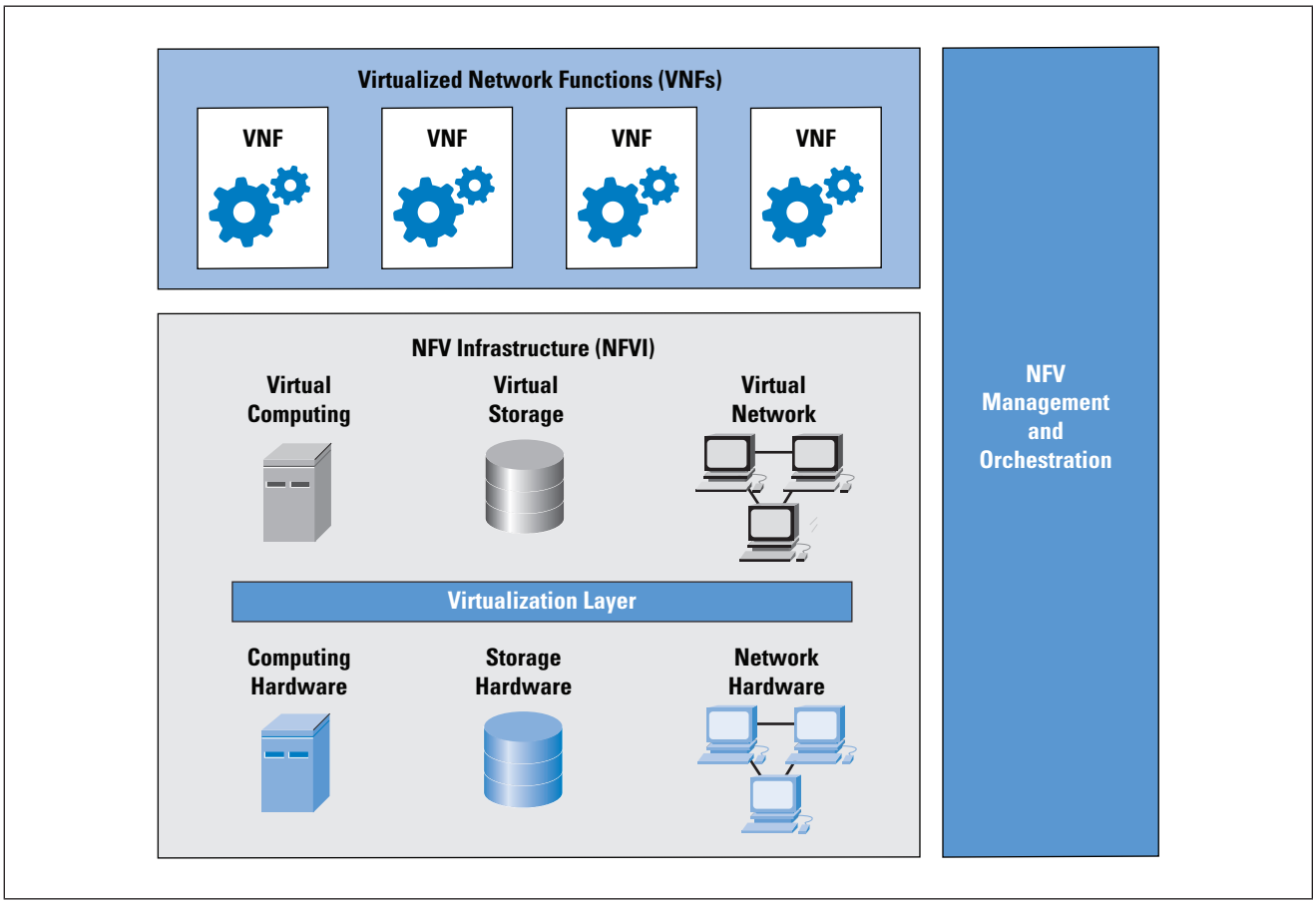
- *Service Chaining:* VNFs are modular and each VNF provides limited functionality on its own. For a given traffic flow within a given application, the service provider steers the flow through multiple VNFs to achieve the desired network functions. This practice is referred to as *service chaining*.
- *Management and Orchestration* (MANO): This feature involves deploying and managing the lifecycle of VNF instances. Examples of functions are VNF instance creation, VNF service chaining, monitoring, relocation, shutdown, and billing. MANO also manages the NFV infrastructure elements.
- *Distributed Architecture:* A VNF may be made up of one or more *VNF Components* (VNFC), each of which implements a subset of the VNF functions. Each VNFC may be deployed in one or multiple instances. These instances may be deployed on separate, distributed hosts in order to provide scalability and redundancy.

Figure 2 shows a high-level view of the NFV framework defined by ISG NFV. This framework supports the implementation of network functions as software-only VNFs. Figure 2 provides an overview of the NFV architecture, which is examined in more detail subsequently.

The NFV framework consists of three domains of operation:

- *Virtualized Network Functions:* These functions are a collection of VNFs, implemented in software, that run over the NFVI.
- *NFV Infrastructure* (NFVI): The NFVI performs a virtualization function on the three main categories of devices in the network service environment: computer devices, storage devices, and network devices.
- *MANO:* This function encompasses the orchestration and lifecycle management of physical and/or software resources that support the infrastructure virtualization and lifecycle management of VNFs. NFV management and orchestration focuses on all virtualization-specific management tasks necessary in the NFV framework.

Figure 2: High-Level NFV Framework



The ISG NFV Architectural Framework document specifies that in the deployment, operation, management, and orchestration of VNFs two types of relations between VNFs are supported:

- *VNF Forwarding Graph* (VNF-FG): Covers the case where network connectivity between VNFs is specified, such as a chain of VNFs on the path to a web server tier (for example, firewall, network address translator, or load balancer).
- *VNF Set:* Covers the case where the connectivity between VNFs is not specified, such as a Web server pool.

NFV Reference Architecture

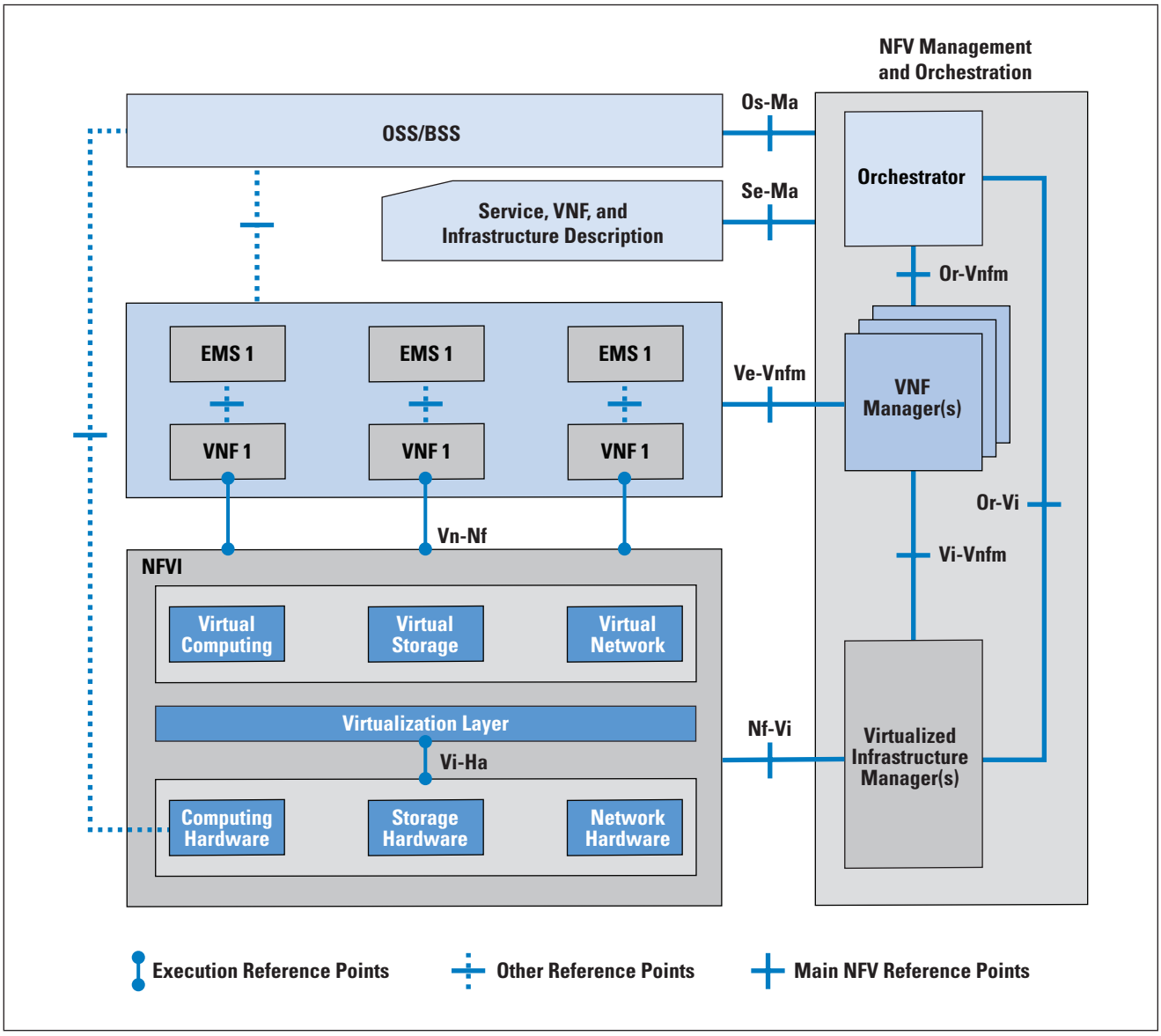
Figure 3 shows a more detailed look at the ISG NFV reference architectural framework.

The architecture consists of four major blocks:

- *NFV Infrastructure* (NFVI): This block comprises the hardware and software resources that create the environment in which VNFs are deployed. NFVI virtualizes physical computing, storage, and networking and places them into resource pools.

- *VNF/EMS:* This collection of VNFs is implemented in software to run on virtual computing, storage, and networking resources, together with a collection of element management systems that manage the VNFs.
- *NFV Management and Orchestration* (NFV-MANO): This framework manages and orchestrates all resources in the NFV environment, including computing, networking, storage, and VM resources
- *Operational and Business Support Systems* (OSS/BSS): The NFV service provider implements this system.

Figure 3: NFV Reference Architectural Framework



It also is useful to view the architecture as consisting of three layers. The NFVI together with the virtualized infrastructure manager provides and manages the virtual resource environment and its underlying physical resources.

The VNF layer provides the software implementation of network functions, together with element management systems and one or more VNF managers. Finally, there is a management, orchestration, and control layer consisting of OSS/BSS and the NFV orchestrator.

NFV Management and Orchestration

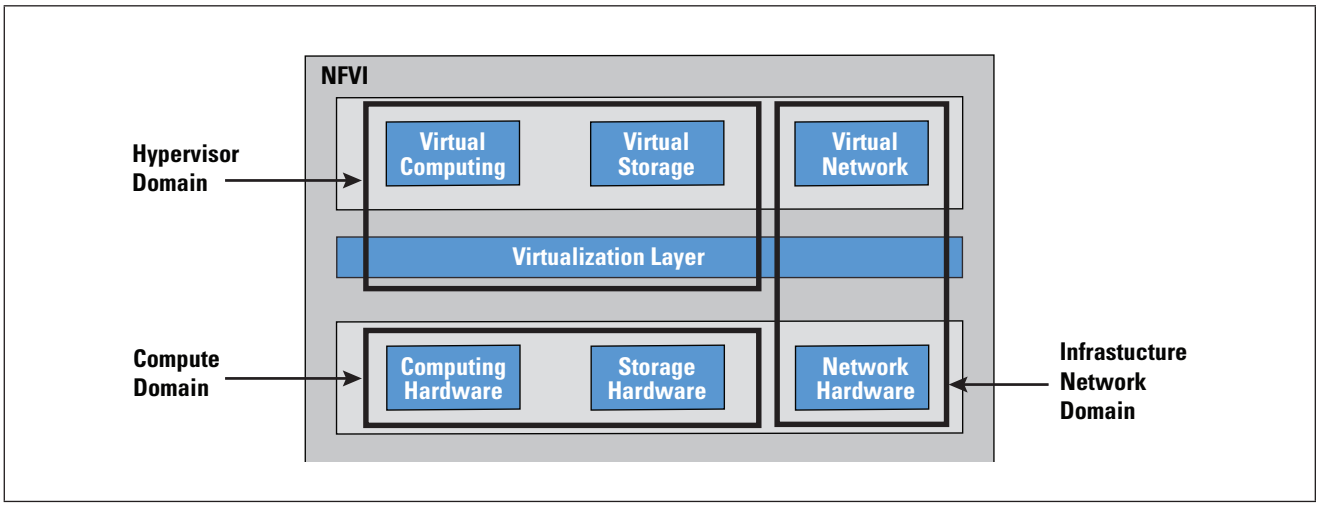
The NFV management and orchestration facility includes the following functional blocks:

- *NFV Orchestrator:* Responsible for installing and configuring new *Network Services* (NS) and VNF packages; NS lifecycle management; global resource management; and validation and authorization of NVFI resource requests.
- *VNF Manager:* Oversees lifecycle management of VNF instances.
- *Virtualized Infrastructure Manager:* Controls and manages the interaction of a VNF with computing, storage, and network resources under its authority, as well as their virtualization.

NFV Infrastructure

The heart of the NFV architecture is a collection of resources and functions known as the *NFV Infrastructure* (NFVI). The NFVI encompasses three domains (Figure 4):

Figure 4: NFV Domains

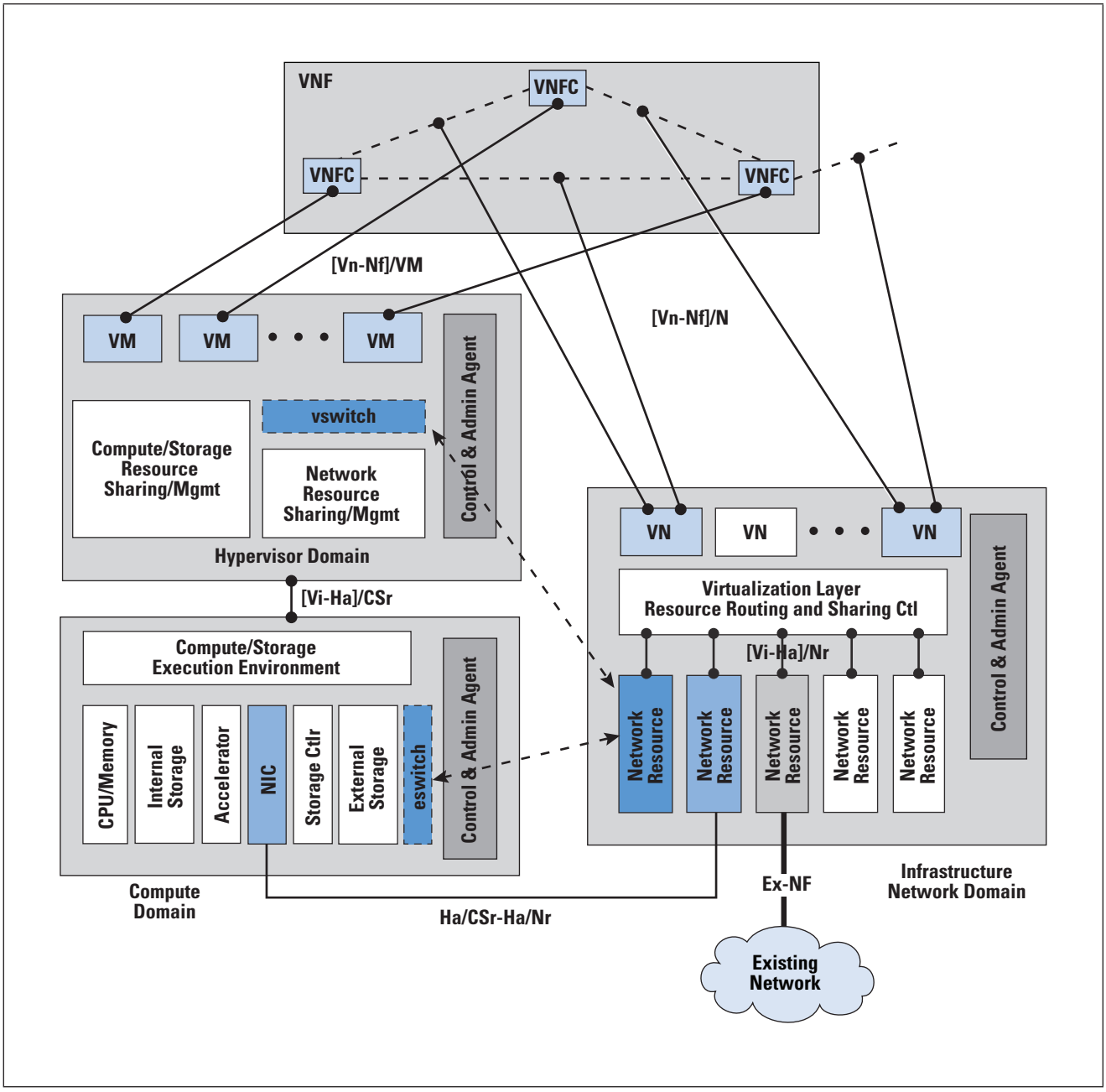


- *Compute Domain:* Provides COTS high-volume servers and storage
- *Hypervisor Domain:* Mediates the resources of the compute domain to the VMs of the software appliances, providing an abstraction of the hardware
- *Infrastructure Network Domain:* Comprises all the generic high-volume switches interconnected into a network that you can configure to supply infrastructure network services.

Logical Structure of NFVI Domains

The ISG NFV standards documents lay out the logical structure of the NFVI domains and their interconnections. The specifics of the actual implementation of the elements of this architecture will evolve in both open-source and proprietary implementation efforts. The NFVI domain logical structure provides a framework for such development and identifies the interfaces between the main components, as shown in Figure 5.

Figure 5: Logical Structure of NFVI Domains



Compute Domain

The principal elements in a typical compute domain may include the following:

- *CPU/Memory:* A COTS processor, with main memory, that executes the VNFC code
- *Internal Storage:* Non-volatile storage housed in the same physical structure as the processor, such as flash memory
- *Accelerator:* Accelerator functions for security, networking, and packet processing
- *External Storage with Storage Controller:* Access to secondary memory devices
- *Network Interface Card* (NIC): An adapter circuit board installed in a computer to provide a physical connection to a network; it provides the physical interconnection with the infrastructure network domain
- *Control & Admin Agent:* Connects to the *Virtualized Infrastructure Manager* (VIM); see Figure 2
- *eswitch:* Server-embedded switch; the eswitch function, described in the following paragraph, is implemented in the compute domain, but functionally it forms an integral part of the infrastructure network domain
- *Compute/Storage Execution Environment:* The execution environment that the server or storage device presents to the hypervisor software

To understand the functions of the eswitch, first note that broadly speaking, VNFs deal with two different kinds of workloads: control plane and data plane. Control-plane workloads are concerned with signaling and control-plane protocols such as the *Border Gateway Protocol* (BGP). Typically, these workloads are more processor- than I/O-intensive, and they do not place a significant burden on the I/O system. Data-plane workloads are concerned with the routing, switching, relaying, or processing of network traffic payloads. Such workloads can require high I/O throughput.

In a virtualized environment such as NFV, all VNF network traffic would go through a virtual switch in the hypervisor domain, which invokes a layer of software between virtualized VNF software and host networking hardware. This situation can create a significant performance penalty. The purpose of the eswitch is to bypass the virtualization software and provide the VNF with a *Direct Memory Access* (DMA) path to the NIC. The eswitch approach accelerates packet processing without any processor overhead.

Hypervisor Domain

The hypervisor domain is a software environment that abstracts hardware and implements services, such as starting a VM, terminating a VM, acting on policies, scaling, live migration, and high availability. The principal elements in the hypervisor domain follow:

- *Compute/storage Resource Sharing/Management:* This service manages these resources and provides virtualized resource access for VMs.
- *Network Resource Sharing/Management:* This service manages these resources and provides virtualized resource access for VMs.
- *Virtual Machine Management and Application Programming Interface* (API): This service provides the execution environment of a single VNFC instance.
- *Control & Admin Agent:* This agent connects to the *Virtualized Infrastructure Manager* (VIM); see Figure 3.
- *vswitch:* The vswitch function, described in the following paragraph, is implemented in the hypervisor domain. However, functionally it forms an integral part of the infrastructure network domain.

The vswitch is an Ethernet switch implemented by the hypervisor that interconnects virtual NICs of VMs with each other and with the NIC of the compute node. If two VNFs are on the same physical server, they are connected through the same vswitch. If two VNFs are on different servers, the connection passes through the first vswitch to the NIC and then to an external switch. This switch forwards the connection to the NIC of the desired server. Finally, this NIC forwards it to its internal vswitch and then to the destination VNF.

Infrastructure Network Domain

The *Infrastructure Network Domain* (IND) performs numerous roles. It provides:

- The communication channel between the VNFCs of a distributed VNF
- The communications channel between different VNFs
- The communication channel between VNFs and their orchestration and management
- The communication channel between components of the NFVI and their orchestration and management
- The means of remote deployment of VNFCs
- The means of interconnection with the existing carrier network

An important distinction is to be made between the virtualization function provided by the hypervisor domain and that provided by the infrastructure network domain. Virtualization in the hypervisor domain uses VM technology to create an execution environment for individual VNFCs.

Virtualization in IND creates virtual networks for interconnection of VNFCs with each other and with network nodes outside the NFV ecosystem. These latter types of nodes are called *Physical Network Functions* (PNFs).

Virtualized Network Functions

A VNF is a virtualized implementation of a traditional network function. Table 1 contains examples of functions that could be virtualized.

Table 1: Potential Network Functions to Be Virtualized

Network Element	Function
Switching elements	Broadband network gateways, carrier-grade Network Address Translation (NAT), and routers
Mobile network nodes	Home Location Register/Home Subscriber Server, gateway, GPRS support node, radio network controller, and various node B functions
Customer premises equipment	Home routers and set-top boxes
Tunneling gateway elements	*IP Security* (IPSec)/SSL virtual private network gateways
Traffic analysis	*Deep packet inspection* (DPI) and *quality of experience* (QoE) measurement
Assurance	Service assurance, *service-level agreement* (SLA) monitoring, and testing and diagnostics
Signaling	Session border controllers and IP Multimedia Subsystem components
Control plane/access functions	*Authentication, Authorization, and Accounting* (AAA) servers, policy control and charging platforms, and *Dynamic Host Configuration Protocol* (DHCP) servers
Application optimization	Content-delivery networks, cache servers, load balancers, and accelerators
Security	Firewalls, virus scanners, intrusion detection systems, and spam protection
Support for General Topologies (not just DC fabrics)	No

As discussed earlier, a VNF comprises one or more *VNF Components* (VNFCs). The VNFCs of a single VNF are connected internal to the VNF. This internal structure is not visible to other VNFs or to the VNF user. An important property of VNFs is elasticity, which means being able to perform one or more of the following:

- *Scale up:* Expand capability by adding resources to a single physical machine or virtual machine.
- *Scale down:* Reduce capability by removing resources from a single physical machine or virtual machine.
- *Scale out:* Expand capability by adding additional physical or virtual machines.
- *Scale in:* Reduce capability by removing physical or virtual machines.

Every VNF has an associated elasticity parameter of no elasticity, scale up/down only, scale out/in only, or both scale up/down and scale out/in.

A VNF is scaled by scaling one or more of its constituent VNFCs. Scale out/in is implemented by adding/removing VNFC instance(s) that belong to the VNF being scaled. Scale up/down is implemented by adding/removing resources from existing VNFC instance(s) that belong to the VNF being scaled.

Summary

NFV provides a powerful, vendor-independent approach to implementing complex networks with dynamic demands. NFV builds on well-established technologies, including virtual machines, containers, and virtual networks. With the demand from 5G and cloud service providers, as well as enterprises with large internal networks, NFV is becoming an increasingly widespread technology.

Further Reading

Greater technical detail is available in many survey papers on NFV.[5, 6, 7, 8] ETSI maintains an NFV web site that includes the ETSI NFV specifications, white papers, tutorials, and a variety of other documents and links (**https://www.etsi.org/technologies/nfv/**). A detailed discussion of the role of NFV in 5G is in [9].

References

[1] ISG NFV, "Network Functions Virtualization: An Introduction, Benefits, Enablers, Challenges & Call for Action," ISG NFV White Paper, October 2012.

[2] Bloomberg L.P., "Network Function Virtualization (NFV) Market Worth $36.3 Billion by 2024," January 15, 2020. **https://www.bloomberg.com/press-releases/2020-01-15/network-function-virtualization-nfv-market-worth-36-3-billion-by-2024-exclusive-report-by-marketsandmarkets**

[3] ISG NFV, "Network Operator Perspectives on NFV Priorities for 5G," ISG NFV White Paper, February 2017.

[4] ITU-T, "Requirements of the IMT-2020 Network," ITU-T Recommendation Y.3101, April 2018.

[5] Mijumbi, R., et al. "Network Function Virtualization: State-of-the-Art and Research Challenges," *IEEE Communications Surveys & Tutorials*, First Quarter, 2016.

[6] Li, Y., and Chen, M. "Software-Defined Network Function Virtualization: A Survey," *IEEE Access*, December 16, 2016.

[7] Li, X., and Qian, C. "A Survey of Network Function Placement." *13th IEEE Annual Consumer Communications & Networking Conference* (CCNC), 2016.

[8] Veeraraghavan, M., et al. "Network Function Virtualization: A Survey," *IEEE Transactions on Communications*, November 2017.

[9] Stallings, W., *5G Wireless: A Comprehensive Introduction,* ISBN-13: 9780136767145, Pearson Education, Inc., 2021.

WILLIAM STALLINGS is an independent consultant and author of numerous books on security, computer networking, and computer architecture. His latest book is *5G Wireless: A Comprehensive Introduction,* (Pearson, 2021). He maintains a computer science resource site for computer science students and professionals at **ComputerScienceStudent.com** and is on the editorial board of Cryptologia. He has a Ph.D. in computer science from M.I.T. He can be reached at: **wllmst@icloud.com**

Our Privacy Policy

The *General Data Protection Regulation* (GDPR) is a regulation for data protection and privacy for all individual citizens of the *European Union* (EU) and the *European Economic Area* (EEA). Its implementation in May 2018 led many organizations worldwide to post or update privacy statements regarding how they handle information collected in the course of business. Such statements tend to be long and include carefully crafted legal language. We realize that we may need to provide similar language on our website and in the printed edition, but until such a statement has been developed here is an explanation of how we use any information you have supplied relating to your subscription:

- The mailing list for *The Internet Protocol Journal* (IPJ) is entirely "opt in." We never have and never will use mailing lists from other organizations for any purpose.
- You may unsubscribe at any time using our online subscription system or by contacting us via e-mail. We will honor any request to remove your name and contact information from our database.
- We will use your contact information only to communicate with you about your subscription; for example, to inform you that a new issue is available, that your subscription needs to be renewed, or that your printed copy has been returned to us as undeliverable by the postal authorities.
- We will never use your contact information for any other purpose or provide the subscription list to any third party other than for the purpose of distributing IPJ by post or by electronic means.
- If you make a donation in support of the journal, your name will be listed on our website and in print unless you tell us otherwise.

Fragments

Workshop: Measuring Network Quality for End-Users

The *Internet Architecture Board* (IAB) is organizing a virtual workshop, September 14–16, 2021. The Internet in 2021 is quite different from what it was 10 years ago. Today, it is a crucial part of everyone's daily life. People use the Internet for their social life, for their daily jobs, for routine shopping, and for keeping up with major events. An increasing number of people can access a Gigabit connection, which would be hard to imagine a decade ago. And, thanks to improvements in security, people trust the Internet for both planning their finances and for everyday payments.

At the same time, some aspects of end-user experience have not improved as much. Many users have typical connection latency that remains at decade-old levels. Despite significant reliability improvements in data center environments, end users often see interruptions in service. Transport refinements, such as QUIC, Multipath TCP, and TCP Fast Open are still not fully supported in some networks. Likewise, various advances in the security and privacy of user data are not widely supported, such as encrypted DNS to the local resolver. We believe that one of the major factors behind this lack of progress is the popular perception that throughput is often the sole measure of the quality of Internet connectivity. With such narrow focus, people don't consider questions such as:

- What is the latency under typical working conditions?
- How reliable is the connectivity across longer time periods?
- Does the network allow the use of a broad range of protocols?
- What services can be run by clients of the network?
- What kind of IPv4, NAT or IPv6 connectivity is offered, and are there firewalls?
- What security mechanisms are available for local services, such as DNS?
- To what degree are the privacy, confidentiality, integrity and authenticity of user communications guarded?

Improving these aspects of network quality will likely depend on measurement and exposing metrics to all involved parties, including to end users in a meaningful way. Such measurements and exposure of the right metrics will allow service providers and network operators to focus on the aspects that impacts the users' experience most and at the same time empowers users to choose the Internet service that will give them the best experience. The IAB is holding this workshop to convene interested researchers, network operators, and Internet technologists to share their experiences and to collaborate on the steps needed to define properties and metrics with the goal of improving Internet access for all users. The workshop will discuss the following questions:

- What are the fundamental properties of a network that contribute to good user experience?
- What metrics quantify these properties, and how to collect such metrics in a practical way?
- What are the best practices for interpreting those metrics, and incorporating those in a decision-making process?
- What are the best ways to communicate these properties to service providers and network operators?
- How can these metrics be displayed to users in a meaningful way?

We realize that the answers to these questions will vary depending on the different experiences of the participants. For example, a commercial video-streaming platform may prioritize higher throughput and to rely on latency-hiding techniques, while a massive multiplayer online game may prioritize lower jitter, and invest into techniques for graceful degradation of the user experience in case of reduced network capacity. At the same time, researchers from the academia may be looking at properties and metrics that haven't been adopted by the industry at all. Likewise, participants may endorse different methodologies for interpreting the metrics and for making decisions. We are actively looking for identifying such methodologies and for capturing the respective best practices.

While this workshop isn't focusing on the solution space, we are welcoming submissions that dive into particular technologies, to the extent of helping to set the context for the discussion. Comparing the merits of specific solutions, however, is outside of the workshop's scope. Interested participants are invited to submit position papers on the workshop questions. Paper size is not limited, but brevity is encouraged. Interested participants who have published relevant academic papers may submit these as a position paper, optionally with a short abstract. The workshop itself will be a virtual meeting over several sessions, with focused discussion based on the position paper topics received. The logistics for the workshop is as follows:

- Submissions Due: August 2, 2021, midnight AOE (Anywhere On Earth)
- Invitations Issued by: August 16, 2021
- Workshop Dates: September 14–16, 2021 (1400–1800 UTC each day)
- Send Submissions to: **network-quality-workshop-pc@iab.org**

The Program Committee members are Jari Arkko, Olivier Bonaventure, Vint Cerf, Stuart Cheshire, Sam Crowford, Nick Feamster, Jim Gettys, Toke Høiland-Jørgensen, Geoff Huston, Cullen Jennings, Mirja Kuehlewind, Jason Livingood, Matt Mathias, Randall Meyer, Kathleen Nichols, Christoph Paasch, Tommy Pauly, Greg White, and Keith Winstein. The workshop co-chairs are Wes Hardaker, Eugeny Khorov, and Omer Shapira.

Position papers from academia, industry, the open source community and others that focus on measurements, experiences, observations and advice for the future are welcome. Papers that reflect experience based on deployed services are especially welcome. The organizers understand that specific actions taken by operators are unlikely to be discussed in detail, so papers discussing general categories of actions and issues without naming specific technologies, products, or other players in the ecosystem are expected. Papers should not focus on specific protocol solutions. The workshop will be by invitation only. Those wishing to attend should submit a position paper to the address above; it may take the form of an Internet-Draft.

All inputs submitted and considered relevant will be published on the workshop website. The organizers will decide whom to invite based on the submissions received. Sessions will be organized according to content, and not every accepted submission or invited attendee will have an opportunity to present as the intent is to foster discussion and not simply to have a sequence of presentations. Position papers from those not planning to attend the virtual sessions themselves are also encouraged. A workshop report will be published afterwards.

For more information, see:
`https://www.iab.org/activities/workshops/network-quality/`

The APNIC Foundation

The *Asia Pacific Network Information Centre* (APNIC) and the *APNIC Foundation* share a common vision of "a global, open, stable, and secure Internet that serves the entire Asia Pacific community." Under its charter, the Foundation seeks to "advance education, on a non-profit making basis, in technical, operational and policy matters relating to Internet infrastructure, through undertaking or funding activities in Hong Kong and elsewhere in the Asia and the Pacific region."

Incorporated in September 2016 and operational in early 2017, the Foundation was first discussed by the APNIC *Executive Council* (EC) in 2014, when it set out to explore a mechanism to support and expand the APNIC Development Program. The EC wanted to do this by raising funds, independent from APNIC membership contributions, to support regional Internet development efforts in the future

Projects and activities funded by the Foundation are designed and managed by APNIC, in collaboration with funding partners interested in Internet development. These activities are implemented by APNIC and our partners, which include a growing group of community trainers and technical advisors, and other like-minded organizations.

The Foundation is guided by an independent Board of Directors—selected by the APNIC EC—that includes recognized and respected experts from the Asia Pacific Internet community.

The Foundation's staff are based in the APNIC office in Brisbane, Australia. The Foundation welcomes support from, and collaboration with, other foundations, agencies and organizations working to develop the Internet in the Asia Pacific.

With more than 13,000 direct and indirect Members in almost every economy of the Asia Pacific, APNIC has spent over 20 years supporting the Internet to serve the region's 3 billion citizens. Many of its 80-plus staff travel regularly in the region to support Members, provide training and technical assistance, or share expertise and information. APNIC also partners with many organizations through MoUs, sponsorships and informally to support the continuing development of the Internet. APNIC's success in partnering and seeking financial support for its activities is founded on five important assets:

- A strong technical focus and regional recognition as a source of best practice and expertise.
- Neutrality and independence from any particular vendors, services, or technologies.
- A non-profit organization with financial strength and transparency.
- Robust regional networks and relationships.
- Long track record of successful management and implementation.

The APNIC Foundation builds on and supports these strengths and APNIC's strong history of success in training and community development.

APNIC development partners have included the Australian *Department of Foreign Affairs and Trade* (DFAT); Canada's *International Development Research Centre* (IDRC); the *Swedish International Development Cooperation Agency* (Sida); the *Japan International Cooperation Agency* (JICA); the World Bank; the United Nations' *International Telecommunications Union* (ITU), the *Internet Corporation for Assigned Names and Numbers* (ICANN), the DotAsia Organization, and the Internet Society. For more information, visit: **https://apnic.foundation/**

EU Launches COVID Certificate

In June 2021, the *European Union* (EU) announced the *Digital Green Certificate*, also known as "Corona Pass" or *Digital COVID Certificate* (DCC), to certify that a European resident has been vaccinated, has recently received a COVID test, or has recovered from the COVID-19 virus. The certificate is used to facilitate travel within EU, and in some cases to allow entrance to some large indoor events. The certificate itself is a QR code, and the majority of its components rely on standards developed by the *Internet Engineering Task Force* (IETF). Éric Vynce explains the details in a blog post here:

http://evyncke.blogspot.com/2021/06/open-source-standards-at-rescue-to.html

Thank You!

Publication of IPJ is made possible by organizations and individuals around the world dedicated to the design, growth, evolution, and operation of the global Internet and private networks built on the Internet Protocol. The following individuals have provided support to IPJ. You can join them by visiting **http://tinyurl.com/IPJ-donate**

Kjetil Aas
Fabrizio Accatino
Michael Achola
Martin Adkins
Melchior Aelmans
Christopher Affleck
Scott Aitken
Jacobus Akkerhuis
Antonio Cuñat Alario
Nicola Altan
Marcelo do Amaral
Matteo D'Ambrosio
Selva Anandavel
Jens Andersson
Danish Ansari
Finn Arildsen
Tim Armstrong
Richard Artes
Michael Aschwanden
David Atkins
Jac Backus
Jaime Badua
Bent Bagger
Eric Baker
Santosh Balagopalan
Benjamin Barkin Wilkins
Michael Bazarewsky
David Belson
Hidde Beumer
Pier Paolo Biagi
Tyson Blanchard
John Bigrow
Orvar Ari Bjarnason
Axel Boeger
Keith Bogart
Mirko Bonadei
Roberto Bonalumi
Julie Bottorff Photography
Gerry Boudreaux
L de Braal
Kevin Breit
Thomas Bridge
Ilia Bromberg
Václav Brožík
Christophe Brun
Gareth Bryan
Stefan Buckmann
Caner Budakoglu
Darrell Budic
BugWorks
Scott Burleigh
Chad Burnham
Jon Harald Bøvre
Olivier Cahagne
Antoine Camerlo
Tracy Camp
Ignacio Soto Campos
Fabio Caneparo
Roberto Canonico
David Cardwell
John Cavanaugh
Lj Cemeras
Dave Chapman
Stefanos Charchalakis
Greg Chisholm
David Chosrova
Marcin Cieslak
Lauris Cikovskis
Guido Coenders
Brad Clark
Narelle Clark
Horst Clausen
Joseph Connolly
Steve Corbató
Brian Courtney
Beth and Steve Crocker
Dave Crocker
Kevin Croes
John Curran
André Danthine
Morgan Davis
Jeff Day
Julien Dhallenne
Freek Dijkstra
Geert Van Dijk
David Dillow
Richard Dodsworth
Ernesto Doelling
Michael Dolan
Eugene Doroniuk
Karlheinz Dölger
Joshua Dreier
Lutz Drink
Dmitriy Dudko
Andrew Dul
Joan Marc Riera Duocastella
Pedro Duque
Holger Durer
Mark Eanes
Andrew Edwards
Peter Robert Egli
George Ehlers
Peter Eisses
Torbjörn Eklöv
Y Ertur
ERNW GmbH
ESdatCo
Steve Esquivel
Jay Etchings
Mikhail Evstiounin
Bill Fenner
Paul Ferguson
Ricardo Ferreira
Kent Fichtner
Armin Fisslthaler
Michael Fiumano
The Flirble Organisation
Gary Ford
Jean-Pierre Forcioli
Susan Forney
Christopher Forsyth
Andrew Fox
Craig Fox
Fausto Franceschini
Valerie Fronczak
Tomislav Futivic
Laurence Gagliani
Edward Gallagher
Andrew Gallo
Chris Gamboni
Xosé Bravo Garcia
Osvaldo Gazzaniga
Kevin Gee
Greg Giessow
John Gilbert
Serge Van Ginderachter
Greg Goddard
Tiago Goncalves
Ron Goodheart
Octavio Alfageme Gorostiaga
Barry Greene
Jeffrey Greene
Richard Gregor
Martijn Groenleer
Geert Jan de Groot
Christopher Guemez
Gulf Coast Shots
Sheryll de Guzman
Rex Hale
Jason Hall
James Hamilton
Stephen Hanna
Martin Hannigan
John Hardin
David Harper
Edward Hauser
David Hauweele
Marilyn Hay
Headcrafts SRLS
Hidde van der Heide
Johan Helsingius
Robert Hinden
Asbjørn Højmark
Damien Holloway
Alain Van Hoof
Edward Hotard
Bill Huber
Hagen Hultzsch
Kevin Iddles
Mika Ilvesmaki
Karsten Iwen
David Jaffe
Ashford Jaggernauth
Martijn Jansen
Jozef Janitor
John Jarvis
Dennis Jennings
Edward Jennings
Aart Jochem
Brian Johnson
Curtis Johnson
Richard Johnson
Jim Johnston
Jonatan Jonasson
Daniel Jones
Gary Jones
Jerry Jones
Anders Marius Jørgensen
Amar Joshi
Javier Juan
David Jump
Merike Kaeo
Andrew Kaiser
Christos Karayiannis
Daniel Karrenberg
David Kekar
Stuart Kendrick
Robert Kent
Jithin Kesavan
Jubal Kessler
Shan Ali Khan
Nabeel Khatri
Dae Young Kim
William W. H. Kimandu
John King
Russell Kirk
Gary Klesk
Anthony Klopp
Henry Kluge
Michael Kluk
Andrew Koch
Ia Kochiashvili
Carsten Koempe
Richard Koene
Alexader Kogan
Antonin Kral
Robert Krejčí
Mathias Körber
John Kristoff
Terje Krogdahl
Bobby Krupczak
Murray Kucherawy
Warren Kumari
George Kuo
Dirk Kurfuerst
Darrell Lack
Andrew Lamb
Richard Lamb
Yan Landriault
Edwin Lang
Sig Lange
Markus Langenmair
Fred Langham
Tracy LaQuey Parker
Jose Antonio Lazaro Lazaro
Rick van Leeuwen
Simon Leinen
Robert Lewis
Christian Liberale
Martin Lillepuu
Roger Lindholm
Link Light Networks
Sergio Loreti
Eric Louie

Adam Loveless
Guillermo a Loyola
Hannes Lubich
Dan Lynch
Sanya Madan
Miroslav Madić
Alexis Madriz
Carl Malamud
Jonathan Maldonado
Michael Malik
Tarmo Mamers
Yogesh Mangar
Bill Manning
Harold March
Vincent Marchand
Gabriel Marroquin
David Martin
Jim Martin
Ruben Tripiana Martin
Timothy Martin
Carles Mateu
Juan Jose Marin Martinez
Ioan Maxim
David Mazel
Miles McCredie
Brian McCullough
Joe McEachern
Alexander McKenzie
Jay McMaster
Mark Mc Nicholas
Carsten Melberg
Kevin Menezes
Bart Jan Menkveld
Sean Mentzer
William Mills
David Millsom
Desiree Miloshevic
Joost van der Minnen
Thomas Mino
Rob Minshall
Wijnand Modderman
Mohammad Moghaddas
Roberto Montoya
Charles Monson
Andrea Montefusco
Fernando Montenegro
Joel Moore
John More
Maurizio Moroni
Brian Mort
Soenke Mumm
Tariq Mustafa
Stuart Nadin
Michel Nakhla
Mazdak Rajabi Nasab
Krishna Natarajan
Naveen Nathan
Darryl Newman
Thomas Nikolajsen
Paul Nikolich
Travis Northrup
Marijana Novakovic
David Oates
Ovidiu Obersterescu
Tim O'Brien
Mike O'Connor
Mike O'Dell
John O'Neill
Jim Oplotnik
Packet Consulting Limited
Carlos Astor Araujo Palmeira
Alexis Panagopoulos
Gaurav Panwar
Manuel Uruena Pascual
Ricardo Patara
Dipesh Patel
Alex Parkinson
Craig Partridge
Dan Paynter
Leif Eric Pedersen
Rui Sao Pedro
Juan Pena
Chris Perkins
Michael Petry
Alexander Peuchert
David Phelan
Derrell Piper
Rob Pirnie
Marc Vives Piza
Jorge Ivan Pincay Ponce
Victoria Poncini
Blahoslav Popela
Andrew Potter
Eduard Llull Pou
Tim Pozar
David Raistrick
Priyan R Rajeevan
Balaji Rajendran
Paul Rathbone
William Rawlings
Mujtiba Raza Rizvi
Bill Reid
Petr Rejhon
Robert Remenyi
Rodrigo Ribeiro
Glenn Ricart
Justin Richards
Rafael Riera
Mark Risinger
Fernando Robayo
Gregory Robinson
Ron Rockrohr
Carlos Rodrigues
Magnus Romedahl
Lex Van Roon
Alessandra Rosi
David Ross
William Ross
Boudhayan Roychowdhury
Carlos Rubio
Rainer Rudigier
Timo Ruiter
RustedMusic
Babak Saberi
George Sadowsky
Scott Sandefur
Sachin Sapkal
Arturas Satkovskis
PS Saunders
Richard Savoy
John Sayer
Phil Scarr
Gianpaolo Scassellati
Elizabeth Scheid
Jeroen Van Ingen Schenau
Carsten Scherb
Ernest Schirmer
Philip Schneck
Peter Schoo
Dan Schrenk
Richard Schultz
Timothy Schwab
Roger Schwartz
SeenThere
Scott Seifel
Yury Shefer
Yaron Sheffer
Doron Shikmoni
Tj Shumway
Jeffrey Sicuranza
Thorsten Sideboard
Greipur Sigurdsson
Andrew Simmons
Pradeep Singh
Henry Sinnreich
Geoff Sisson
Helge Skrivervik
Terry Slattery
Darren Sleeth
Richard Smit
Bob Smith
Courtney Smith
Eric Smith
Mark Smith
Tim Sneddon
Craig Snell
Job Snijders
Ronald Solano
Asit Som
Ignacio Soto Campos
Evandro Sousa
Peter Spekreijse
Thayumanavan Sridhar
Paul Stancik
Ralf Stempfer
Matthew Stenberg
Adrian Stevens
Clinton Stevens
John Streck
Martin Streule
David Strom
Viktor Sudakov
Edward-W. Suor
Vincent Surillo
Terence Charles Sweetser
T2Group
Roman Tarasov
David Theese
Douglas Thompson
Kerry Thompson
Lorin J Thompson
Fabrizio Tivano
Joseph Toste
Rey Tucker
Sandro Tumini
Angelo Turetta
Phil Tweedie
Steve Ulrich
Unitek Engineering AG
John Urbanek
Martin Urwaleck
Betsy Vanderpool
Surendran Vangadasalam
Ramnath Vasudha
Philip Venables
Buddy Venne
Alejandro Vennera
Luca Ventura
Scott Vermillion
Tom Vest
Dario Vitali
Jeffrey Wagner
Don Wahl
Michael L Wahrman
Laurence Walker
Randy Watts
Andrew Webster
Tim Weil
Jd Wegner
Westmoreland Engineering Inc.
Rick Wesson
Peter Whimp
Russ White
Jurrien Wijlhuizen
Derick Winkworth
Pindar Wong
Phillip Yialeloglou
Janko Zavernik
Bernd Zeimetz
Muhammad Ziad Ziayuddin
Tom Zingale
Jose Zumalave
Romeo Zwart
廖 明沂.

Call for Papers

The *Internet Protocol Journal* (IPJ) is a quarterly technical publication containing tutorial articles ("What is...?") as well as implementation/operation articles ("How to..."). The journal provides articles about all aspects of Internet technology. IPJ is not intended to promote any specific products or services, but rather is intended to serve as an informational and educational resource for engineering professionals involved in the design, development, and operation of public and private internets and intranets. In addition to feature-length articles, IPJ contains technical updates, book reviews, announcements, opinion columns, and letters to the Editor. Topics include but are not limited to:

- Access and infrastructure technologies such as: Wi-Fi, Gigabit Ethernet, SONET, xDSL, cable, fiber optics, satellite, and mobile wireless.
- Transport and interconnection functions such as: switching, routing, tunneling, protocol transition, multicast, and performance.
- Network management, administration, and security issues, including: authentication, privacy, encryption, monitoring, firewalls, troubleshooting, and mapping.
- Value-added systems and services such as: Virtual Private Networks, resource location, caching, client/server systems, distributed systems, cloud computing, and quality of service.
- Application and end-user issues such as: E-mail, Web authoring, server technologies and systems, electronic commerce, and application management.
- Legal, policy, regulatory and governance topics such as: copyright, content control, content liability, settlement charges, resource allocation, and trademark disputes in the context of internetworking.

IPJ will pay a stipend of US$1000 for published, feature-length articles. For further information regarding article submissions, please contact Ole J. Jacobsen, Editor and Publisher. Ole can be reached at **ole@protocoljournal.org** or **olejacobsen@me.com**

Supporters and Sponsors

Supporters

Internet Society

CISCO

Diamond Sponsors

Your logo here!

Ruby Sponsors

ICANN

Sapphire Sponsors

APNIC FOUNDATION

Emerald Sponsors

Afilias

Akamai

APRICOT Asia Pacific Regional Internet Conference on Operational Technologies

COMCAST

DE-CIX

EQUINIX

Google

JPRS JAPAN REGISTRY SERVICES

JUNIPER NETWORKS

lacnic

LinkedIn

linx

netskope

NSRC Network Startup Resource Center

NTT Communications

RIPE NCC RIPE NETWORK COORDINATION CENTRE

TEAM CYMRU

VERISIGN

WIDE PROJECT

Corporate Subscriptions

AFRINIC The Internet Numbers Registry for Africa

amsix amsterdam internet exchange

DNS-OARC

IWL

ISC Internet Systems Consortium

IPXO

Limelight NETWORKS

qa|cafe

For more information about sponsorship, please contact **sponsor@protocoljournal.org**

The Internet Protocol Journal
Link Fulfillment
7650 Marathon Dr., Suite E
Livermore, CA 94550

CHANGE SERVICE REQUESTED

The Internet Protocol Journal is published quarterly and supported by the Internet Society and other organizations and individuals around the world dedicated to the design, growth, evolution, and operation of the global Internet and private networks built on the Internet Protocol.

Email: ipj@protocoljournal.org
Web: www.protocoljournal.org

Printed in the USA on recycled paper.