UNITED STATES DISTRICT COURT
WESTERN DISTRICT OF TEXAS
WACO DIVISION

Intellectual Ventures I LLC and Intellectual Ventures II LLC,	)	Civil Action No. 6:21-cv-226
Plaintiff,	)	
v.	)	
Hewlett Packard Enterprise Company,	)	
Defendant.	)	**JURY TRIAL DEMANDED**

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiffs, Intellectual Ventures I LLC and Intellectual Ventures II LLC (together "IV"), for their complaint against defendant, Hewlett Packard Enterprise Company ("HPE"), hereby allege as follows:

THE PARTIES

1. Intellectual Ventures I LLC ("Intellectual Ventures I") is a Delaware limited liability company having its principal place of business located at 3150 139th Avenue SE, Bellevue, Washington 98005.

2. Intellectual Ventures II LLC ("Intellectual Ventures II") is a Delaware limited liability company having its principal place of business located at 3150 139th Avenue SE, Bellevue, Washington 98005.

3. Upon information and belief, HPE is a Delaware corporation with its principal executive offices located at 11445 Compaq Center West Drive, Houston, Texas 77070. HPE has regular and established places of business in this District, including a fifty-two (52) acre campus

at 14321 Tandem Boulevard, Austin, Texas, and a lease for another 27,326 square foot office at Paloma Ridge, 13620 FM 620 Austin, Texas 78717. HPE also has at least one other office in Texas, at 6080 Tennyson Parkway, Suite 400, Plano, Texas 75024. HPE plans to relocate its global headquarters from San Jose, California, to Spring, Texas, in early 2022. HPE may be served with process through its registered agent, CT Corporation System, 1999 Bryan Street, Suite 900, Dallas, Texas 75201.

JURISDICTION

4. IV brings this action for patent infringement under the United States' patent laws, 35 U.S.C. § 271, *et seq.* This Court has subject matter jurisdiction under 28 U.S.C. §§ 1331 and 1338(a).

5. This Court has general jurisdiction over HPE because HPE is engaged in substantial and not isolated activity at its regular and established places of business within this judicial district. This Court has specific jurisdiction over HPE because HPE has committed acts of infringement giving rise to this action within this judicial district and has established more than minimum contacts within this judicial district, such that the exercise of jurisdiction over HPE in this Court would not offend traditional notions of fair play and substantial justice.

6. Venue is proper in this judicial district pursuant to 28 U.S.C. §§ 1391(b)-(c) and 1400(b) because HPE maintains regular and established places of business and has committed acts of patent infringement within this judicial district.

FACTUAL BACKGROUND

7. Intellectual Ventures Management, LLC ("Intellectual Ventures") was founded in 2000. Since then, Intellectual Ventures has been involved in the invention business. Intellectual Ventures fosters inventions and facilitates the filing of patent applications for those inventions;

collaborates with others to develop and patent inventions; and acquires and licenses patents from individual inventors, universities, corporations, and other institutions. A significant aspect of Intellectual Ventures' business is managing the plaintiffs in this case, Intellectual Ventures I and Intellectual Ventures II.

8. One founder of Intellectual Ventures is Nathan Myhrvold, who worked at Microsoft from 1986 until 2000 in a variety of executive positions, culminating in his appointment as the company's first Chief Technology Officer ("CTO") in 1996. While at Microsoft, Dr. Myhrvold founded Microsoft Research in 1991 and was one of the world's foremost software experts. Between 1986 and 2000, Microsoft became the world's largest technology company.

9. Under Dr. Myhrvold's leadership, IV acquired more than 70,000 patents covering many important inventions of the Internet-era. Many of these inventions coincided with Dr. Myhrvold's successful tenure at Microsoft.

10. Two significant accomplishments of the Internet era are the related technologies of cloud computing and virtualization. Cloud computing enables ubiquitous access to shared pools of configurable computing system resources, such as memory and storage, as well as software programs that execute atop them, such as desktop applications and websites. These resources and programs, often collectively referred to as "the cloud," can be rapidly provisioned with minimal effort over the Internet. Virtualization allows for the creation of multiple software environments of dedicated computer resources that each simulate for its users a computing environment that was traditionally physically distinct and close to a user, such as an environment provided by a computer server and/or a connected array of disk/flash drives. This has enabled computing services to be delivered to a given customer using centralized computing resources that seem as though they are

located just a few feet away from the user, even if they are actually located hundreds or even thousands of miles away (hence the terms "virtual private service" and "virtual private server").

11. A significant consequence of cloud computing and virtualization has been the migration of computing from isolated environments (e.g., an office) centered around physical private servers to distributed systems implemented on large numbers of virtual private servers running inside the cloud. As a result, a very large number of virtual private servers have been deployed, often to the cloud, and each require a secure and stable file system. File systems that were known at the turn of this century, however, proved insufficient for these sorts of new high-frequency deployments as they required copying the entire file system of a host computer for use by each virtual private server, thus wasting an immense amount of storage and compute power or exposing users to security flaws by insufficiently isolating one user's data from another user's data. They also could not be efficiently backed up. Therefore, there was a need for file systems that could serve many virtual private servers without requiring extensive copying or wasted storage space and that could be efficiently backed up.

12. Another consequence of cloud computing and virtualization has been the proliferation of distributed computing platforms in which a number of systems (be they physical or virtual) are managed, controlled and coordinated to work on a distributed project. Previously existing distributed computing platforms did not optimally manage, control and coordinate the various systems comprising such a platform as they worked on a distributed project, such as a network site testing project. While such a platform could work on a project by statically assigning various portions of the project to different systems, it was not possible to dynamically manage and control those systems in a coordinated way, for instance, by automatically increasing the number of systems involved as they were working on the project in response to some pre-determined

condition that impacts the desired processing power assigned to the project. Such a shortcoming was problematic, for example, when attempting to work on a project that required unexpected and extreme surges in processing, such as surges in computation required to test a network site using a simulated Denial of Service attack (DoS).

13. The aforementioned advances in computing have also caused a massive increase in the frequency of computer input/output (I/O) operations, such as read/write storage operations or networking operations supporting voice connections. Importantly, the frequency of I/O operations typically vary tremendously depending on a multiplicity of factors, including the time of day/week. Previously existing I/O systems were assigned to a single group of users, for example the co-located employees of a company, and could easily get overloaded at certain times of day, such as the start of business when many users are reading/writing emails or placing VoIP calls. It was not feasible to overprovision those I/O systems to handle peak loads, given the relatively small number of I/O requests the systems would need to handle most of the time. There was a need for storage, networking, and other I/O resources that could be flexibly arranged to more cost-effectively support varying loads of I/O operations or be dynamically provisioned in real-time responsive to varying loads and compute/network conditions under policy driven user controls.

14. More and more of the aforementioned compute resources, including those made available using cloud and virtualization technologies, have been accessed by users of wireless devices through relatively low-cost Wi-Fi network access points. Those users have been congregating in large numbers within limited spaces (i.e., at workplaces, apartment buildings, public facilities such as transport hubs, etc.) while attempting to concurrently initiate and maintain high bandwidth connections into Wi-Fi networks. The resulting network traffic was overwhelming existing Wi-Fi networks in large part because the congregated Wi-Fi devices tended to interfere

with one another, even when they were part of different networks. There was, therefore, also a need for Wi-Fi connections that could be flexibly and automatically adjusted to better accommodate large numbers of closely orientated Wi-Fi users.

15. HPE makes, uses, and sells compute, computer storage and computer networking products and services, with a particular focus on providing products and services using cloud computing and virtualization. Three of the most important technology areas that HPE focuses on to deliver these cloud-centric and virtualized compute products and services are: (1) virtual private servers; (2) resilient I/O systems that cost-effectively handle varying loads of I/O requests even as compute/network conditions fluctuate; and (3) advanced Wi-Fi access points that can provide high-bandwidth and cost-effective access to compute products and services to many closely orientated users. When providing virtual private servers through its GreenLake or Ezmeral offerings for example, HPE provides a form of virtual private server called a Docker container, that uses a file system that is optimized to minimize extensive copying, wasted storage space, and inefficient backups. HPE also enables orchestration of activities by these Docker containers, using Kubernetes-based technologies. When providing I/O systems through its 3PAR storage or Aruba Wide Area Network (WAN) Optimization offerings for example, HPE makes them more resilient and cost-effective by either virtualizing the I/O resources provided by the offerings, and/or enabling access to those I/O resources to be automatically configured responsive to changes in usage and network conditions. When providing Wi-Fi access points through its Aruba Wi-Fi offerings, HPE is focused on allowing its customers to flexibly and automatically adjust the operating parameters of those access points to accommodate large numbers of closely separated users.

THE PATENTS-IN-SUIT

16. On September 9, 2003, the Patent & Trademark Office ("PTO") issued United States Patent No. 6,618,736 ("the '736 patent"), titled TEMPLATE-BASED CREATION AND ARCHIVAL OF FILE SYSTEMS. The '736 patent is valid and enforceable. A copy of the '736 patent is attached as Exhibit A.

17. Intellectual Ventures I is the owner and assignee of all rights, title and interest in and to the '736 patent, and holds all substantial rights therein, including the rights to grant licenses, to exclude others, and to enforce and recover past damages for infringement of that patent.

18. The inventions claimed in the '736 patent were conceived while its inventor, Paul Menage, worked at Ensim Corporation. Dr. Menage is highly respected in his field with over 20 years of experience at companies such as Ensim, Google, and Facebook, to name a few. Dr. Menage holds a bachelor's degree and a Ph.D. in computer science from the University of Cambridge. Dr. Menage has published several papers and articles on containerization, resource management and virtualization throughout his career and played integral roles in implementing solutions in his capacity at Google and Facebook.

19. The '736 patent is directed to systems, methods and/or apparatus for an improved way to create, manage and archive file systems. For instance, through the use of shared and private storage units correlated via a usage map that is particularly useful in a virtualized environment. The claimed approach allows for the creation and management of separate file systems for a plurality of virtual private servers, running on a physical host computer ("host"), that are vying for the resources of that host, without requiring extensive copying or wasted storage space. This in turn enables users to gain more storage capability from their existing hosts without needing to overprovision them, and therefore, without needing to purchase access to additional hardware storage resources for their existing virtual private servers. It also enables a more efficient back-up

of a file system of a virtual private server. Preexisting file systems could not deliver these outcomes.

20. On February 15, 2011, the PTO issued United States Patent No. RE 42,153 ("the '153 patent"), titled DYNAMIC COORDINATION AND CONTROL OF NETWORK CONNECTED DEVICES FOR LARGE-SCALE NETWORK SITE TESTING AND ASSOCIATED ARCHITECTURES. The '153 patent is valid and enforceable. A copy of the '153 patent is attached as Exhibit B.

21. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the '153 patent, and holds all substantial rights therein, including the rights to grant licenses, to exclude others, and to enforce and recover past damages for infringement of that patent.

22. The inventions of the '153 patent were developed by Edward A. Hubbard, Krishnamurthy Venkatramani, David P. Anderson, Ashok K. Adiga, Greg D. Hewgill, and Jeff A. Lawson. Dr. Anderson is a research scientist at the Space Sciences Laboratory at U.C. Berkeley and an adjunct professor of computer science at the University of Houston. He helped create and now leads the SETI@home software project, developed the first distributed system for digital audio editing, and served as CTO of United Devices which developed software for distributed computing systems in the early 2000s. Jeff A. Lawson is a software engineer with extensive experience in software development, including with respect to distributed computing. Early in his career, Mr. Lawson worked for NASA's Jet Propulsion Laboratory as a software developer working on image acquisition software deployed aboard NASA Space Shuttles. Mr. Lawson, like Mr. Anderson, previously worked at United Devices, designing, architecting, implementing, and supporting highly scalable distributed computing software for enterprise customers. He also co-

founded distributed.net, a large-scale distributed computing network of over 50,000 computers worldwide. The other co-inventors of the '153 patent have similarly illustrious credentials.

23. The '153 patent is directed to systems, methods and/or apparatus for the dynamic coordination and control of network connected systems that collectively perform distributed processing projects such as the testing of a network site. More specifically, in the novel architecture covered by the '153 patent, distributed processing of a project collectively occurs on a plurality of systems (referenced in the patent as "client systems") that each participate in the project in large part by executing a client agent program. Throughout such distributed project processing, poll communications from client systems are received at a managing system (referenced in the patent as a "server system") to enable it to form a dynamic snapshot of current overall project status. Analysis of the dynamic snapshot status information is then performed by the server system to determine if it should decrease or increase the number of client systems that are actively participating in the project, and corresponding poll response communications are sent from the server to the client systems. The poll communications and poll response communications are used by the server system to dynamically coordinate the ongoing project processing performed by the client systems. The '153 patent's system differs from prior art systems for example, by using status snapshots about a project generated from poll communications to dynamically control and coordinate project activities occurring across the client systems as those activities are occurring, rather than for example using such snapshots and poll communications for use in future resource planning well after those project activities have terminated. The'153 patented system thus covers a far more dynamic way to adjust a distributed processing system, in response to any required change in processing requirements.

24. On August 24, 2010, the PTO issued United States Patent No. 7,783,788 ("the '788 patent"), titled VIRTUAL INPUT/OUTPUT SERVER. The '788 patent is valid and enforceable. A copy of the '788 patent is attached as Exhibit C.

25. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the '788 patent, and holds all substantial rights therein, including the rights to grant licenses, to exclude others, and to enforce and recover past damages for infringement of the '788 patent.

26. The inventions claimed in the '788 patent were conceived while the inventors worked at 3Leaf Systems, Inc., a pioneer in the network virtualization field, and include Robert Quinn, co-founder of 3Leaf Systems. The company received over $67M in funding, partnering with industry giants like IBM and Intel to develop network virtualization solutions that by one account are "changing the way storage and server virtualization are done." Prior to its acquisition by Huawei, 3Leaf was granted numerous patents, including the '788 patent.

27. The '788 patent is directed to systems, methods and/or apparatus for virtualizing I/O subsystems, such as storage or networking subsystems. As one example, this novel approach allows for multiple application servers (whether physical or virtual) to share an I/O subsystem by virtualizing the resources of that I/O subsystem, receiving configuration information relating to the I/O subsystem, and then regulating usage of the virtualized I/O subsystem by each of the servers through configuring utilization of a physical interface associated with the I/O subsystem. One non-limiting benefit of this approach is that it increases the flexibility with which I/O resources can be used by servers, thus gaining increased performance of existing systems and without subjecting shared virtualized resources to excessive usage by one of more of the servers. This was not possible with preexisting systems.

28. On March 25, 2014, the PTO issued United States Patent No. RE 44,818 ("the RE '818 patent"), titled QUALITY OF SERVICE IN VIRTUAL COMPUTING ENVIRONMENTS. The RE '818 patent is valid and enforceable. A copy of the RE '818 patent is attached as Exhibit D.

29. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the RE '818 patent and holds all substantial rights therein, including the right to grant licenses, to exclude others, and to enforce and recover past damages for infringement of the RE '818 patent.

30. The inventions claimed in the RE '818 patent were conceived while the inventors worked at 3Leaf Systems, Inc., like the inventors of the aforementioned '788 patent. The inventors of the RE '818 patent went on to work at several prominent companies such as Google.

31. Similar to the '788 patent, the RE '818 patent is also directed to systems, methods and/or apparatus for the operation of virtualized input/output (I/O) systems in environments where access to such systems by virtual or physical application servers involves virtual interfaces. As virtualization's popularity and use increased, virtualized I/O systems had to handle a greater number of communications ("virtual I/O communications"), from a greater number of servers, and—as a result of several technological advances including those captured by the aforementioned '788 patent—those I/O systems had to support a wider range of quality of service (QoS) requirements. The RE '818 helps address those requirements by enabling, for instance, configuration of usage parameters for a physical interface associated with the virtual I/O subsystems to occur in an increasingly sophisticated manner that could be based on more than one type of variable, such as the customer requesting the virtual I/O communication or the type of operation being supported by the virtual I/O communication.

32. On November 9, 2004, the PTO issued United States Patent No. 6,816,464 ("the '464 patent"), titled METHOD, SYSTEM, AND COMPUTER PROGRAM PRODUCT FOR ROUTE QUALITY CHECKING AND MANAGEMENT. The '464 patent is valid and enforceable. A copy of the '464 patent is attached as Exhibit E.

33. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the '464 patent, and holds all substantial rights therein, including the rights to grant licenses, to exclude others, and to enforce and recover past damages for infringement of the '464 patent.

34. The inventions claimed in the '464 patent were conceived while the inventors worked at Array Telecom Corporation ("Array"), a wholly owned subsidiary of Comdial Corporation ("Comdial"). Comdial sold and marketed sophisticated communications products and advanced phone systems for small and medium sized businesses. At the time Comdial acquired Array, it noted that "the principal asset purchased was the intellectual property" which enabled Comdial to "become a leading provider of personal computer ("PC")-based voice processing systems and telephony gateways for routing voice and fax communications over the private intranet and the public internet."

35. The '464 patent is directed to systems, methods and/or apparatus for an improved way to establish communication links for exchanging loss-sensitive, delay-sensitive, and/or jitter-sensitive traffic within a network, such as a wide area network (WAN). As more such traffic began traversing WANs, and as user expectations about the quality of such traffic increased, users could no longer rely on traditional network optimization techniques, which suffered from a host of deficiencies. Examples of such deficiencies include, without limitation, not taking into account rapidly changing network conditions, not allowing end users to provide their own criteria for

selecting routes in response to the changing conditions, and not allowing route selections to be made on a more granular per-call/connection basis. The inventors of the '464 patent improved on traditional network optimization techniques, for instance, by providing for the testing and scoring of network routes based on end-user related criteria, and on a per call/connection basis. This in turn allowed for improved quality of service and traffic shaping capabilities for each user that was not possible in prior art solutions. Embodiments of the '464 inventions further teach operation across wide area packet-switched networks (such as the Internet), inclusion of databases that store and allow for consideration of historical routing information, graphical user interfaces ("GUIs") for accepting user routing preferences, and advanced settings for configuring route measurement properties, timings, and statistical analysis.

36. On September 20, 2011, the PTO issued United States Patent No. 8,023,991 ("the '991 patent"), titled PROGRAM FOR ADJUSTING CHANNEL INTERFERENCE BETWEEN ACCESS POINTS IN A WIRELESS NETWORK. A copy of the '991 patent is attached as Exhibit F.

37. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the '991 patent, and holds all substantial rights therein, including the right to grant licenses, to exclude others, and to enforce and recover past damages for infringement of the '991 patent.

38. The inventions of the '991 patent were conceived by Floyd Backes, co-founder, CTO, and ultimately CEO, of Autocell Labs of Acton, MA, among other named co-inventors. Mr. Backes had previously been vice president of strategy and architecture at Nortel Networks, CTO of the switching division at 3Com (one of the so-called "Big Four" startups at the forefront of the data networking revolution), and a senior engineer and architect in networks and communications at Digital Equipment Corporation. Co-inventor Gary Vacon holds electrical engineering degrees

from Tufts University and MIT and has worked on network hardware and wireless networking technologies at Bell Laboratories, Digital Equipment Corporation, and Autocell Labs (where he served as Chairman). Autocell Labs (founded as Propogate Networks) developed innovative solutions that automatically reduced radio frequency ("RF") interference, and thus ensured robust wireless performance in business and wireless broadband deployments. Companies worldwide had embedded Autocell software into their products to seamlessly address the need for RF control and optimization. Investors in Autocell included Chestnut Partners, Inc., Motorola Ventures, and Siemens Venture Capital.

39. The '991 patent is directed to systems, methods and/or apparatus for an improved way to manage wireless communications environments, by issuing instructions to Wi-Fi access points ("APs") that are using an RF channel to adjust their transmit power levels, so as to decrease the radio interference they are causing for other APs using that same RF channel. The '991 patent also calls for basing those instructions on indications of transmit power levels received from other APs, including those APs that are using that same RF channel. The inventions of the '991 patent enable far more APs to operate on the same RF channels while being located in close proximity with one another without provoking excessive interference. This, in turn, allows Wi-Fi network operators to greatly increase the number of end-user stations supported in their networks, and the amount of bandwidth provided to those end-user stations.

40. On May 13, 2014, the PTO issued United States Patent No. 8,725,132 ("the '132 patent"), titled PROGRAM FOR ADJUSTING CHANNEL INTERFERENCE BETWEEN ACCESS POINTS IN A WIRELESS NETWORK. A copy of the '132 patent is attached as Exhibit G.

41. Intellectual Ventures II is the owner and assignee of all rights, title and interest in and to the ’132 patent, and holds all substantial rights therein, including the right to grant licenses, to exclude others, and to enforce and recover past damages for infringement of the ’132 patent.

42. Like the ‘991 patent, the inventions of the ’132 patent were conceived by, among others, co-inventors Floyd Backes and Gary Vacon, whose accomplishments are set out above.

43. The ’132 patent s directed to systems, methods and/or apparatus for an improved way to manage wireless communications environments, by enabling Wi-Fi APs to select an optimal transmit power level, repeatedly adjusting that power level to reduce interference, and causing an associated device (*e.g.*, an end user station such as a smartphone or laptop computer communicating with that AP), to transmit signals at a power level that is set also to reduce interference. The inventions of the ’132 patent enabled far more APs and end user stations to operate in close proximity to one another without provoking excessive interference. As with the ’991 patent, this in turn allows Wi-Fi network operators to greatly increase the number of end-user stations supported in their networks, and the amount of bandwidth provided to those end-user stations.

COUNT I
(HPE’s Infringement of U.S. Patent No. 6,618,736)

44. Paragraphs 1-43 are reincorporated by reference as if fully set forth herein.

45. The inventions claimed in the ’736 patent, taken alone or in combination, were not well-understood, routine or conventional to one of ordinary skill in the art at the time of the invention. Rather, the ’736 patent claims and teaches, *inter alia*, an improved way to create, manage and archive file systems in virtualized environments, which was not present in the state of the art at the time of the invention. For example, the inventions improved upon then existing file system technology by providing an architecture specific to virtualized environments in which

virtual private servers (now implemented, for example, as Docker containers) could be contending for the resources of a single or limited number of physical host computer(s). The invention further improved on prior art solutions by allowing for certain shared portions of a single file system to be commonly accessed by functionally unrelated virtual private servers, while keeping other private portions of the file system siloed to be accessed only by a limited set of one or more virtual private servers.

46. Instead of having to provide a separate physical file system for each virtual private server, or to duplicate any common or shared portions of the file system for each virtual private server, the inventions of the ’736 patent allowed for the segregation of a file system into shared and private portions via a tiered containerized architecture which could be simultaneously utilized without the unnecessary replication or insecure isolation methods relative to prior art systems.

47. The inventions represented a technical solution to an unsolved technological problem. The written description of the ’736 patent describes, in technical detail, each of the limitations in the claims, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claims differ markedly from what had been performed in the industry prior to the inventions of the ’736 patent. More specifically, the claims of the ’736 patent recite methods and systems for creating and archiving one or more file systems within one or more servers that comprise a first set of storage units, each storage unit of the first set corresponding to a storage unit of a second set, a usage map for indicating which of the second storage units contain valid data, an interception module for intercepting an attempt to write a data item to a first storage unit, a writing module for writing the data to the corresponding second

storage unit, and storing an indication in the usage map that the corresponding second storage unit contains valid data.

48. The system covered by the asserted claims, therefore, differs markedly from the prior systems in use at the time of this invention, which, *inter alia*, lacked the claimed combination of first and second sets of storage units and, for example, did not provide for the interception of write commands or rely on usage maps so as to enable the creation and management of a separate file system for a virtual private server (*e.g.*, a container), running on a physical host computer. Further, the claimed inventions differ from prior art systems by using the claimed architecture to enable more efficient file system snapshotting, and thus more efficient backing up, which otherwise would be more expensive in a virtualized environment.

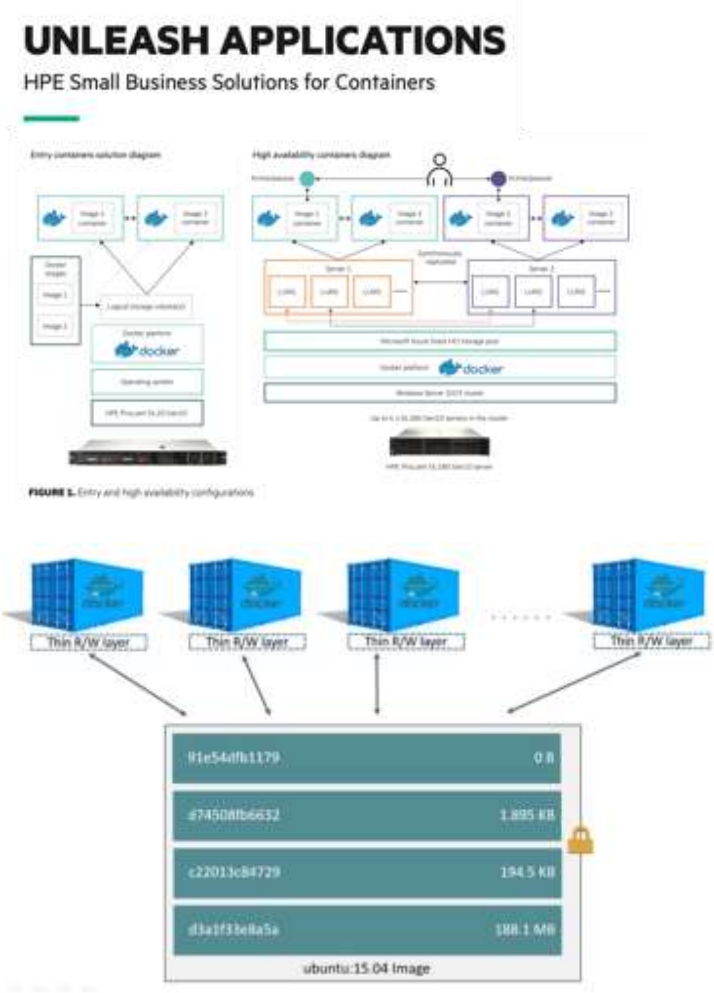
49. As described above, the ’736 patent is drawn to solving a specific, technical problem arising in the context of virtualized computing file system access and management. Consistent with the problem addressed being rooted in such file system access and management environments, the solutions disclosed in the ’736 patent consequently are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

50. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 1 of the ’736 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the ’736 patent. HPE’s products and/or services that infringe the ’736 patent include, but are not limited to, HPE Docker Enterprise Edition, HPE ProLiant family of servers with integrated Docker, HPE Ezmeral Container Platform with Docker integration, HPE GreenLake Service for Containers, and any other HPE products and/or services, either alone or in combination, that operate in substantially the same manner (together the “Accused ’736 Products”).

Claim 1 of the ’736 patent is reproduced below:

> *1. A method for file system creation and archival comprising:*
>
> *providing a first set of storage units and a second set of storage units, each storage unit of the first set corresponding to a storage unit of the second set;*
>
> *providing a first usage map for indicating which storage units of the second set contain valid data;*
>
> *intercepting an attempt to write a data item to a storage unit of the first set;*
>
> *writing the data item to the corresponding storage unit of the second set; and*
>
> *storing an indication in the first usage map that the corresponding storage unit of the second set contains valid data.*

51. The Accused ’736 Products provide a method for creating and archiving file systems. As one non-limiting example, the Accused ’736 Products include Docker Enterprise Edition (EE) and HPE hardware/software with native Docker integration, that create and manage file systems for use by containers:



Because each container has its own writable container layer, and all changes are stored in this container layer, multiple containers can share access to the same underlying image and yet have their own data state. The diagram below shows multiple containers sharing the same Ubuntu 18.04 image.

52. Furthermore, the Accused '736 Products, as integrated with Docker, provide a first set of storage units and a second set of storage units, each storage unit of the first set corresponding to a storage unit of the second set. For instance, each Docker container comprises a series of layers stacked on top of each other, the lower layer containing read-only image data and the upper read/write layer containing data from the lower layer that has been altered during operation of the container:

Images and layers

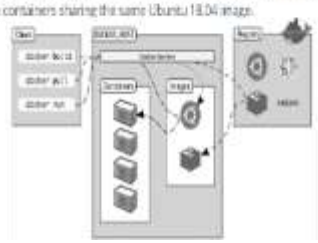
A Docker image is built up from a series of layers. Each layer represents an instruction in the image's Dockerfile. Each layer except the very last one is read-only. Consider the following Dockerfile:

Each layer is only a set of differences from the layer before it. The layers are stacked on top of each other. When you create a new container, you add a new writable layer on top of the underlying layers. This layer is often called the "container layer". All changes made to the running container, such as writing new files, modifying existing files, and deleting files, are written to this thin writable container layer. The diagram below shows a container based on the Ubuntu 15.04 image.

Container and layers

The major difference between a container and an image is the top writable layer. All writes to the container that add new or modify existing data are stored in this writable layer. When the container is deleted, the writable layer is also deleted. The underlying image remains unchanged.

Because each container has its own writable container layer, and all changes are stored in this container layer, multiple containers can share access to the same underlying image and yet have their own data state. The diagram below shows multiple containers sharing the same Ubuntu 15.04 image.



The copy-on-write (CoW) strategy

Copy-on-write is a strategy of sharing and copying files for maximum efficiency. If a file or directory exists in a lower layer within the image, and another layer (including the writable layer) needs read access to it, it just uses the existing file. The first time another layer needs to modify the file (when building the image or running the container), the file is copied into that layer and modified. This minimizes I/O and the size of each of the subsequent layers. These advantages are explained in more depth below.

Docker registries

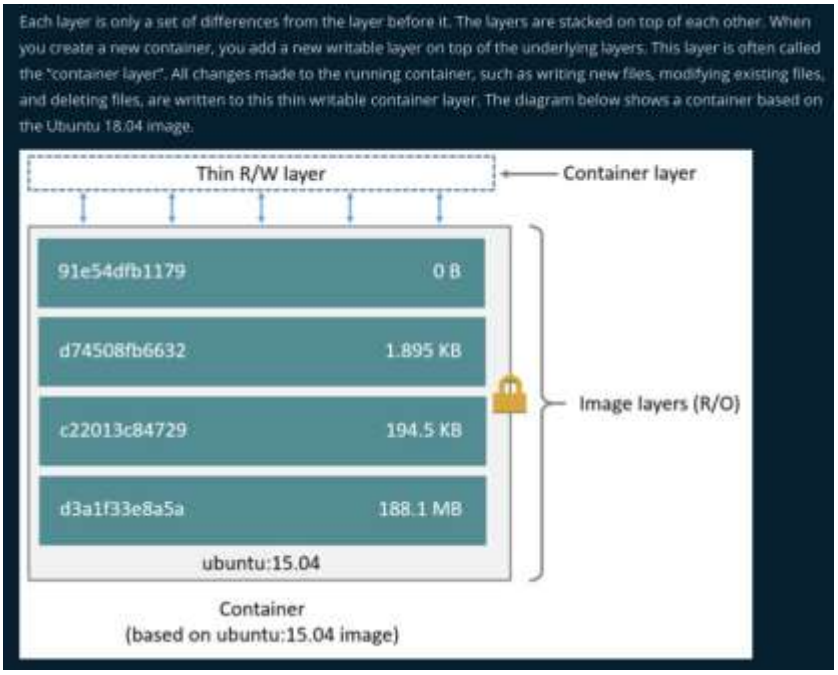
A Docker *registry* stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. You can even run your own private registry.

When you use the `docker pull` or `docker run` commands, the required images are pulled from your configured registry. When you use the `docker push` command, your image is pushed to your configured registry.

IMAGES

An *image* is a read-only template with instructions for creating a Docker container. Often, an image is *based on* another image, with some additional customization. For example, you may build an image which is based on the `ubuntu` image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.

Because each container has its own writable container layer, and all changes are stored in this container layer, multiple containers can share access to the same underlying image and yet have their own data state. The diagram below shows multiple containers sharing the same Ubuntu 18.04 image.



53. The Accused '736 Products, as integrated with Docker, further provide a first usage map for indicating which storage units of the second set contain valid data. For instance, the layered Docker architecture presents a unified view of the image layers and corresponding writable container layer. Through, *e.g.*, a diff directory, as seen below, the system keeps track of which data has been changed and is present in the upper read/write layer. If the data is not present in the upper read/write layer, any attempt to read/write that data will cause it to be copied to the upper layer from the lower layer and tracked by the system:

How the `overlay` driver works

This content applies to the `overlay` driver only. Docker recommends using the `overlay2` driver, which works differently. See How the overlay2 driver works for `overlay2`.

OverlayFS layers two directories on a single Linux host and presents them as a single directory. These directories are called *layers* and the unification process is referred to as a *union mount*. OverlayFS refers to the lower directory as `lowerdir` and the upper directory a `upperdir`. The unified view is exposed through its own directory called `merged`.

The diagram below shows how a Docker image and a Docker container are layered. The image layer is the `lowerdir` and the container layer is the `upperdir`. The unified view is exposed through a directory called `merged` which is effectively the containers mount point. The diagram shows how Docker constructs map to OverlayFS constructs.

How the `overlay2` driver works

If you are still using the `overlay` driver rather than `overlay2`, see How the overlay driver works instead.

OverlayFS layers two directories on a single Linux host and presents them as a single directory. These directories are called *layers* and the unification process is referred to as a *union mount*. OverlayFS refers to the lower directory as `lowerdir` and the upper directory a `upperdir`. The unified view is exposed through its own directory called `merged`.

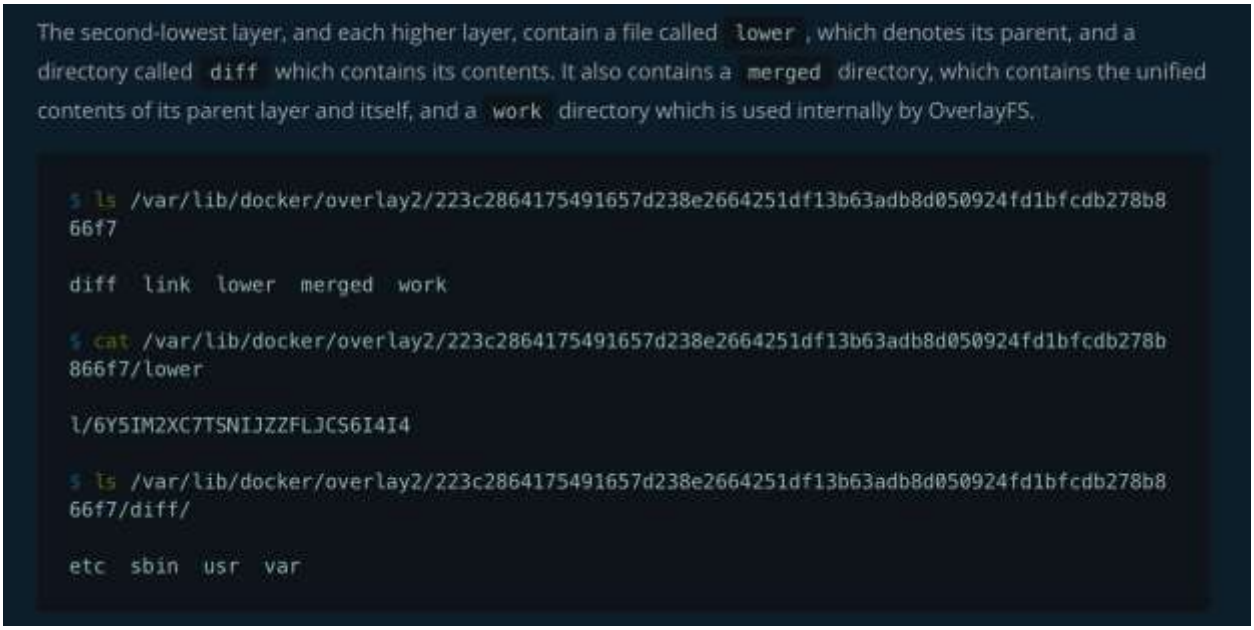
The `overlay2` driver natively supports up to 128 lower OverlayFS layers. This capability provides better performance for layer-related Docker commands such as `docker build` and `docker commit`, and consumes fewer inodes on the backing filesystem.

Copying makes containers efficient

When you start a container, a thin writable container layer is added on top of the other layers. Any changes the container makes to the filesystem are stored here. Any files the container does not change do not get copied to this writable layer. This means that the writable layer is as small as possible.

When an existing file in a container is modified, the storage driver performs a copy-on-write operation. The specifics steps involved depend on the specific storage driver. For the `aufs`, `overlay`, and `overlay2` drivers, the copy-on-write operation follows this rough sequence:

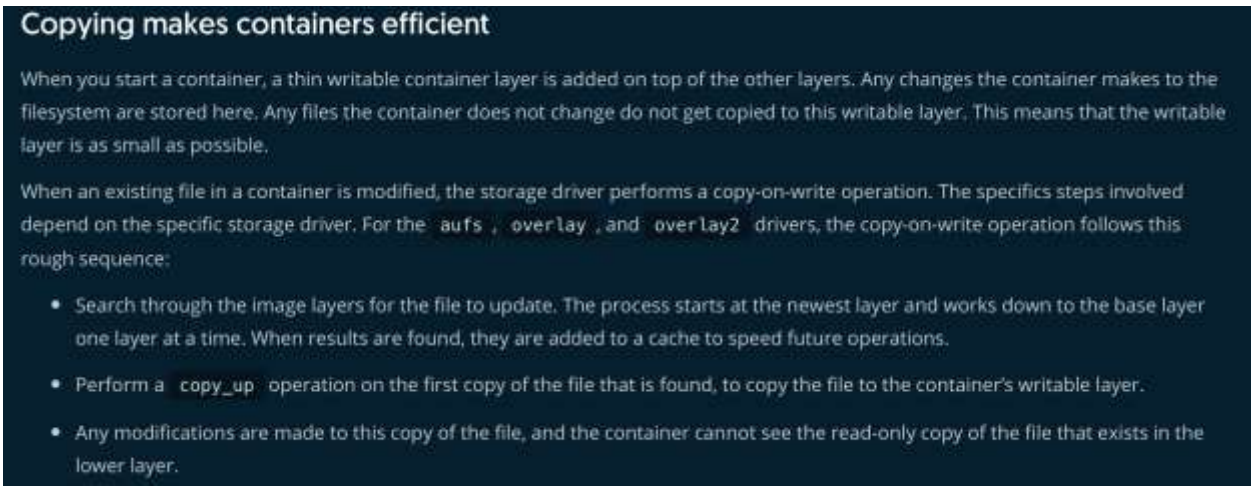
- Search through the image layers for the file to update. The process starts at the newest layer and works down to the base layer one layer at a time. When results are found, they are added to a cache to speed future operations.
- Perform a `copy_up` operation on the first copy of the file that is found, to copy the file to the container's writable layer.
- Any modifications are made to this copy of the file, and the container cannot see the read-only copy of the file that exists in the lower layer.



The second-lowest layer, and each higher layer, contain a file called `lower`, which denotes its parent, and a directory called `diff` which contains its contents. It also contains a `merged` directory, which contains the unified contents of its parent layer and itself, and a `work` directory which is used internally by OverlayFS.

```
$ ls /var/lib/docker/overlay2/223c2864175491657d238e2664251df13b63adb8d050924fd1bfcdb278b8
66f7

diff  link  lower  merged  work

$ cat /var/lib/docker/overlay2/223c2864175491657d238e2664251df13b63adb8d050924fd1bfcdb278b
866f7/lower

l/6Y5IM2XC7TSNIJZZFLJCS6I4I4

$ ls /var/lib/docker/overlay2/223c2864175491657d238e2664251df13b63adb8d050924fd1bfcdb278b8
66f7/diff/

etc  sbin  usr  var
```

54. In addition, the Accused '736 Products intercept an attempt to write a data item to a storage unit of the first set. For example, the Docker engine integrated with the Accused Products uses a "Copy-on-Write" mechanism to write data. When a write of a data item is attempted, rather than writing the item directly to storage units associated with the image layer that contains that data, the write attempt is intercepted, the data is copied from the lower image layer to the upper read/write layer, and the operation is performed on the data in the upper layer, as illustrated below:



Copying makes containers efficient

When you start a container, a thin writable container layer is added on top of the other layers. Any changes the container makes to the filesystem are stored here. Any files the container does not change do not get copied to this writable layer. This means that the writable layer is as small as possible.

When an existing file in a container is modified, the storage driver performs a copy-on-write operation. The specifics steps involved depend on the specific storage driver. For the `aufs`, `overlay`, and `overlay2` drivers, the copy-on-write operation follows this rough sequence:

- Search through the image layers for the file to update. The process starts at the newest layer and works down to the base layer one layer at a time. When results are found, they are added to a cache to speed future operations.
- Perform a `copy_up` operation on the first copy of the file that is found, to copy the file to the container's writable layer.
- Any modifications are made to this copy of the file, and the container cannot see the read-only copy of the file that exists in the lower layer.

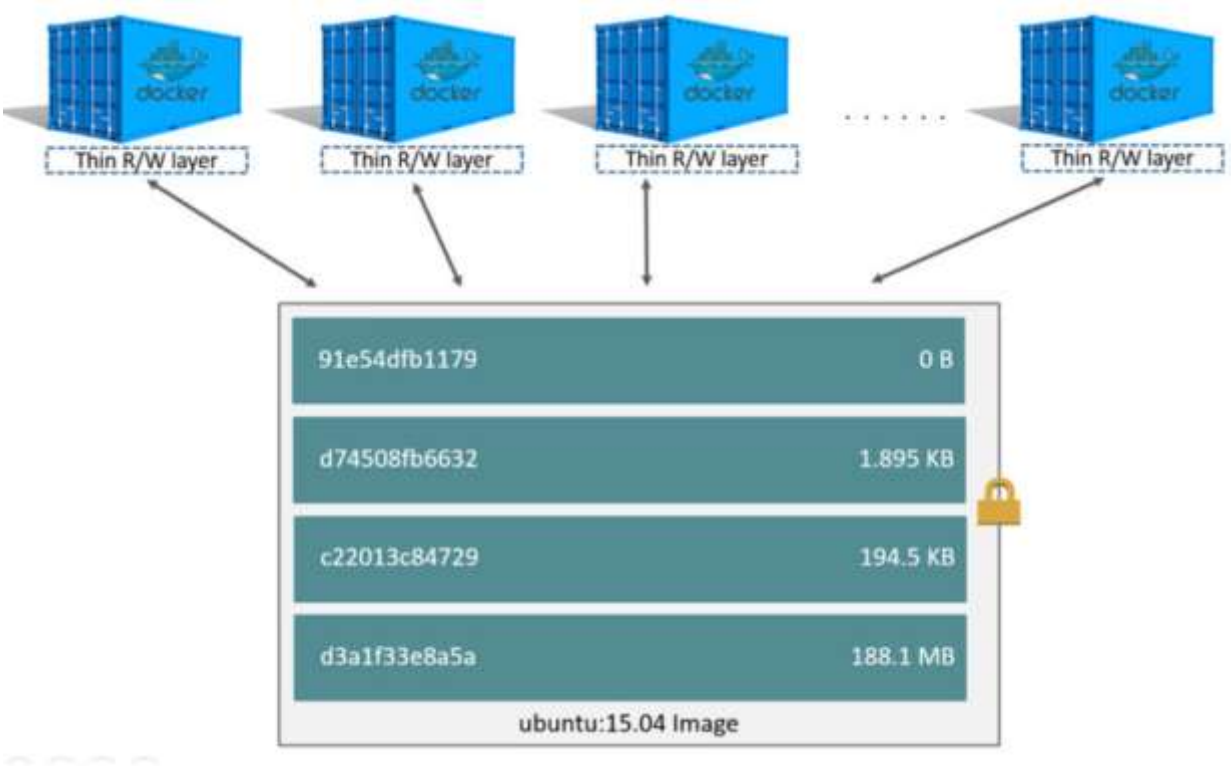
55. The Accused ’736 Products, as integrated with Docker, further write the data item to the corresponding storage unit of the second set. For example, reiterating the “copy-on-write” summary provided just above when a write of a data item to a given file is attempted, the data items of the identified image layer are “copied up” into storage units associated with the writeable container layer and new data items are written to the storage units associated with the writable container layer, all as illustrated below:

The copy-on-write (CoW) strategy

Copy-on-write is a strategy of sharing and copying files for maximum efficiency. If a file or directory exists in a lower layer within the image, and another layer (including the writable layer) needs read access to it, it just uses the existing file. The first time another layer needs to modify the file (when building the image or running the container), the file is copied into that layer and modified. This minimizes I/O and the size of each of the subsequent layers. These advantages are explained in more depth below.

The major difference between a container and an image is the top writable layer. All writes to the container that add new or modify existing data are stored in this writable layer. When the container is deleted, the writable layer is also deleted. The underlying image remains unchanged.

Because each container has its own writable container layer, and all changes are stored in this container layer, multiple containers can share access to the same underlying image and yet have their own data state. The diagram below shows multiple containers sharing the same Ubuntu 18.04 image.



56. The Accused '736 Products additionally store an indication in the first usage map that the corresponding storage unit of the second set contains valid data. For example, once a file is modified by writing new data items to the writable container layer, as described above, in order to correctly present a unified view of that file after the modification, an indication exists that writes were just made into those storage units and that they therefore contain valid data:

How the `overlay` driver works

This content applies to the `overlay` driver only. Docker recommends using the `overlay2` driver, which works differently. See How the overlay2 driver works for `overlay2`.

OverlayFS layers two directories on a single Linux host and presents them as a single directory. These directories are called *layers* and the unification process is referred to as a *union mount*. OverlayFS refers to the lower directory as `lowerdir` and the upper directory a `upperdir`. The unified view is exposed through its own directory called `merged`.

The diagram below shows how a Docker image and a Docker container are layered. The image layer is the `lowerdir` and the container layer is the `upperdir`. The unified view is exposed through a directory called `merged` which is effectively the containers mount point. The diagram shows how Docker constructs map to OverlayFS constructs.

How the `overlay2` driver works

If you are still using the `overlay` driver rather than `overlay2`, see How the overlay driver works instead.

OverlayFS layers two directories on a single Linux host and presents them as a single directory. These directories are called *layers* and the unification process is referred to as a *union mount*. OverlayFS refers to the lower directory as `lowerdir` and the upper directory a `upperdir`. The unified view is exposed through its own directory called `merged`.

The `overlay2` driver natively supports up to 128 lower OverlayFS layers. This capability provides better performance for layer-related Docker commands such as `docker build` and `docker commit`, and consumes fewer inodes on the backing filesystem.

Copying makes containers efficient

When you start a container, a thin writable container layer is added on top of the other layers. Any changes the container makes to the filesystem are stored here. Any files the container does not change do not get copied to this writable layer. This means that the writable layer is as small as possible.

When an existing file in a container is modified, the storage driver performs a copy-on-write operation. The specifics steps involved depend on the specific storage driver. For the `aufs`, `overlay`, and `overlay2` drivers, the copy-on-write operation follows this rough sequence:

- Search through the image layers for the file to update. The process starts at the newest layer and works down to the base layer one layer at a time. When results are found, they are added to a cache to speed future operations.
- Perform a `copy_up` operation on the first copy of the file that is found, to copy the file to the container's writable layer.
- Any modifications are made to this copy of the file, and the container cannot see the read-only copy of the file that exists in the lower layer.

The Copy-on-Write Mechanism

When we launch an image, the Docker engine does not make a full copy of the already stored image. Instead, it uses something called the *copy-on-write* mechanism. This is a standard UNIX pattern that provides a single shared copy of some data, *until the data is modified*.

To do this, changes between the image and the running container are tracked. Just before any write operation is performed in the running container, a copy of the file that would be modified is placed on the writeable layer of the container, and that is where the write operation takes place. Hence the name, "copy-on-write".

If this wasn't happening, each time you launched an image, a full copy of the filesystem would have to be made. This would add time to the startup process and would end up using a lot of disk space.

57. Additionally, HPE has been, and currently is, an active inducer of infringement of the ’736 patent under 35 U.S.C. § 271(b) and contributory infringement of the ’736 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

58. HPE has actively induced, and continues to actively induce, infringement of the ’736 patent by intending that others use, offer for sale, or sell in the United States, products and/or services covered by the ’736 patent, including but not limited to HPE Docker Enterprise Edition, the HPE ProLiant family hardware as integrated with Docker, and the HPE Ezmeral Container Platform as integrated with Docker, the GreenLake Container Service Platform as integrated with Docker, as well as any HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States the foregoing products and/or services that directly infringe the ’736 patent as described above.

59. HPE has contributed to, and continues to contribute to, the infringement of the ’736 patent by others by offering to sell, selling, or otherwise commercially offering products and/or services that, when installed and configured result in a system as intended by HPE, that directly infringe the ’736 patent.

60. HPE knew of the '736 patent, or should have known of the '736 patent, but was willfully blind to its existence. Upon information and belief, HPE has had actual knowledge of the '736 patent since at least as early as the service upon HPE of this Complaint. On information and belief, HPE had actual knowledge of the '736 patent in and around August 18, 2020. On information and belief, HPE was aware of IV's allegations against Arista Networks, Inc. relating to infringement of the '736 patent by Arista's operating systems utilizing native Docker integration. Additionally, HPE was aware of the '736 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the '736 patent.

61. HPE has committed, and continues to commit, affirmative acts that cause infringement of the '736 patent with knowledge of the '736 patent and knowledge or willful blindness that the induced acts constitute infringement of the '736 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customary way as desired and intended by HPE, infringe the '736 patent and/or by directly or indirectly providing instructions on how to use its products and/or services in a manner or configuration that infringes the '736 patent, including those found at the following:

- https://buy.hpe.com/us/en/software/docker-enterprise-edition/docker-enterprise-edition/docker-enterprise-edition/docker-enterprise-edition-from-hpe/p/1009161344
- https://h20195.www2.hpe.com/V2/GetDocument.aspx?docname=a00101842enw&jumpid=in_smb_dm_container&
- https://h20195.www2.hpe.com/V2/GetDocument.aspx?docname=a00101842enw&jumpid=in_smb_dm_container&#
- https://h20195.www2.hpe.com/v2/gethtml.aspx?docname=a00056658enw
- https://psnow.ext.hpe.com/doc/a00008645enw?jumpid=in_lit-psnow-red

- https://community.hpe.com/t5/HPE-Ezmeral-Uncut/HPE-Ezmeral-simplifies-data-processing-and-analysis/ba-p/7095000#.YDk8si1h1bU
- https://assets.ext.hpe.com/is/content/hpedam/a50002683enw
- https://www.hpe.com/us/en/greenlake/container-platform.html

62. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that when used cause the direct infringement of the '736 patent by a third party, and which have no substantial non-infringing uses, or include a separate and distinct component that is especially made or especially adapted for use in infringement of the '736 patent and is not a staple article or commodity of commerce suitable for substantial non-infringing use.

63. As a result of HPE's acts of infringement, IV has suffered and will continue to suffer damages in an amount to be proved at trial.

COUNT II
(HPE's Infringement of U.S. Patent No. RE 42,153)

64. Paragraphs 1-63 are incorporated by reference as if fully set forth herein.

65. The inventions claimed in the '153 patent, taken alone or in combination, were not well-understood, routine, or conventional to one of ordinary skill in the art at the time of the invention. Rather, the '153 patent claims and teaches, *inter alia*, improved dynamic coordination and control architecture for executing projects within a distributed platform that comprises a server system and a plurality of network-connected client systems. For example, dynamic coordination and control of the execution of a project by the network-connected server and client systems is based on poll communications and responses exchanged between the server system and the client systems. The inventions improved upon then existing distributed computing technology by providing for dynamic snapshot information of a project's status determined based upon poll communications and responses involving the server and client systems, and then taking

coordination and control actions related to the ongoing project based on an analysis of the dynamic snapshot information. An example of such coordination and control actions is dynamically increasing or decreasing the amount of client systems actively participating in the project.

66. Instead of statically configuring the coordination and control actions related to such projects before or after the execution of a project for example, the inventions of the '153 patent allowed, *inter alia*, for an exchange of data between the client systems performing a distributed project, and the server system that was managing the client systems and that enabled the dynamic coordination and control of processing activities across the client systems. Through poll communications between the client and server systems, for example, the number of client systems participating in the project could be automatically dynamically increased or decreased, for example.

67. The inventions represent technological solutions to an unsolved technological problem. The specification of the '153 patent describes, in technical detail, each of the limitations in the claims, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also to understand how the non-conventional and non-generic ordered combination of the elements of the claims differ markedly from what had been done in the industry prior to the inventions of the '153 patent. More specifically, for example, the asserted claims of the '153 patent each recite poll communications between server and client systems through a network in a distributed computing platform, and the dynamic analysis and coordination of distributed project activities during project operations that lead to, for example, increases or decreases in the number of client systems participating in the project as the project is still being executed.

68. The system covered by the asserted claims therefore differs markedly from the systems in use prior to this invention, which, *inter alia*, lacked the claimed combination of dynamic interaction between the server and client systems during ongoing project operations. Furthermore, the claimed inventions differ from prior art systems by using dynamic status snapshots about a project generated in part from poll communications to dynamically control and coordinate project activities across the client systems as those activities are occurring, rather than using such snapshots and communications for use in future resource planning relating to a project.

69. As described above, the ’153 patent is drawn to solving a specific, technical problem arising in the context of distributed processing or computing systems. Consistent with the problem addressed being rooted in such complex, distributed approaches to computing, the solutions provided by the ’153 patent are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

70. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 18 of the ’153 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the ’153 patent. HPE’s products and/or services that infringe the ’153 patent include, but are not limited to, the HPE Ezmeral Container Platform (including with pre-integrated HPE Ezmeral Data Fabric) and HPE Apollo Servers, as well as HPE GreenLake cloud services for containers (including services powered by the HPE Ezmeral Container Platform, and run on the HPE Synergy integrated system), and any other of HPE’s products and/or services, either alone or in combination, that operate in substantially the same manner (the “Accused ’153 Products”). The HPE Ezmeral Container Platform, for example, includes, *inter alia*, fully integrated Kubernetes support for pods, tenants, and clusters; a management interface for monitoring, managing,

provisioning, controlling, and scaling resources, applications, clusters, and services; KubeDirector; a managed gateway; GPU support for Kubernetes hosts; a Kubernetes web terminal; and HPE Ezmeral ML Ops. HPE Apollo systems, including the XL170r, XL190r, XL270d, and 4200 Gen10 servers, are optimized for the HPE Ezmeral Container Platform software (together the "Accused '153 Products").[1]

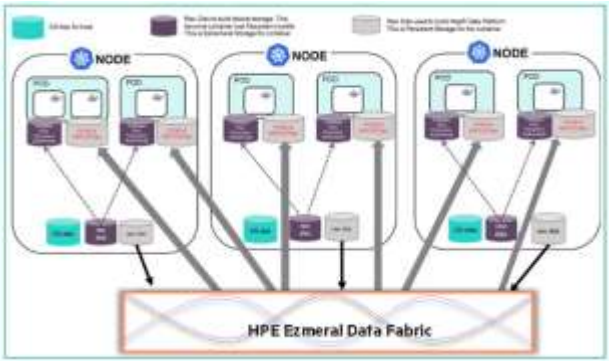
Claim 18 of the '153 patent is reproduced below:

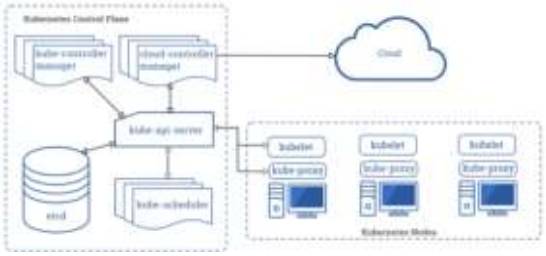
> *18. A distributed computing platform having dynamic coordination capabilities for distributed client systems processing project workloads, comprising:*
> *a plurality of network-connected distributed client systems, the client systems having under-utilized capabilities;*
> *a client agent program configured to run on the client systems and to provide workload processing for at least one project of a distributed computing platform; and*
> *at least one server system configured to communicate with the plurality of client systems through a network to provide the client agent program to the client systems, to send initial project and poll parameters to the client systems, to receive poll communications from the client systems during processing of the project workloads, wherein a dynamic snapshot information of current project status is provided based at least in part upon the poll communications from the client systems, to analyze the poll communications utilizing the dynamic snapshot information to determine whether to change how many client systems are active in the at least one project, wherein if a fewer number is desired, including within a poll response communications a reduction in the number of actively participating clients, and if a greater number is desired, adding client systems to active participation in the at least one project within a poll response communications, the server system repeatedly utilizing the poll communications and the poll response communications to coordinate project activities of the client systems during project operations.*

71. The Accused '153 Products, when deployed on HPE's own servers or those of its customers constitute a distributed computing platform with dynamic coordination capabilities for distributed client systems that are processing project workloads. As one non-limiting example, the HPE Ezmeral Container Platform is a unified container platform for both cloud-native

[1] *See* https://h20195.www2.hpe.com/v2/GetDocument.aspx?docname=a00062186enw#

applications and stateful analytics applications that can run on any infrastructure either on-premises, in public clouds, in a hybrid model, or at the edge. The HPE Ezmeral Container Platform comes with out-of-the-box configuration for Kubernetes orchestration of networking, load-balancing, and storage, and is a certified Kubernetes distribution, with HPE itself considering containers and open-source Kubernetes central to its approach for deploying and managing a containerized environment at scale. The Accused '153 Products dynamically coordinates workload processing across distributed client systems, including by deploying distributed applications such as Apache Spark on Kubernetes across multiple systems:





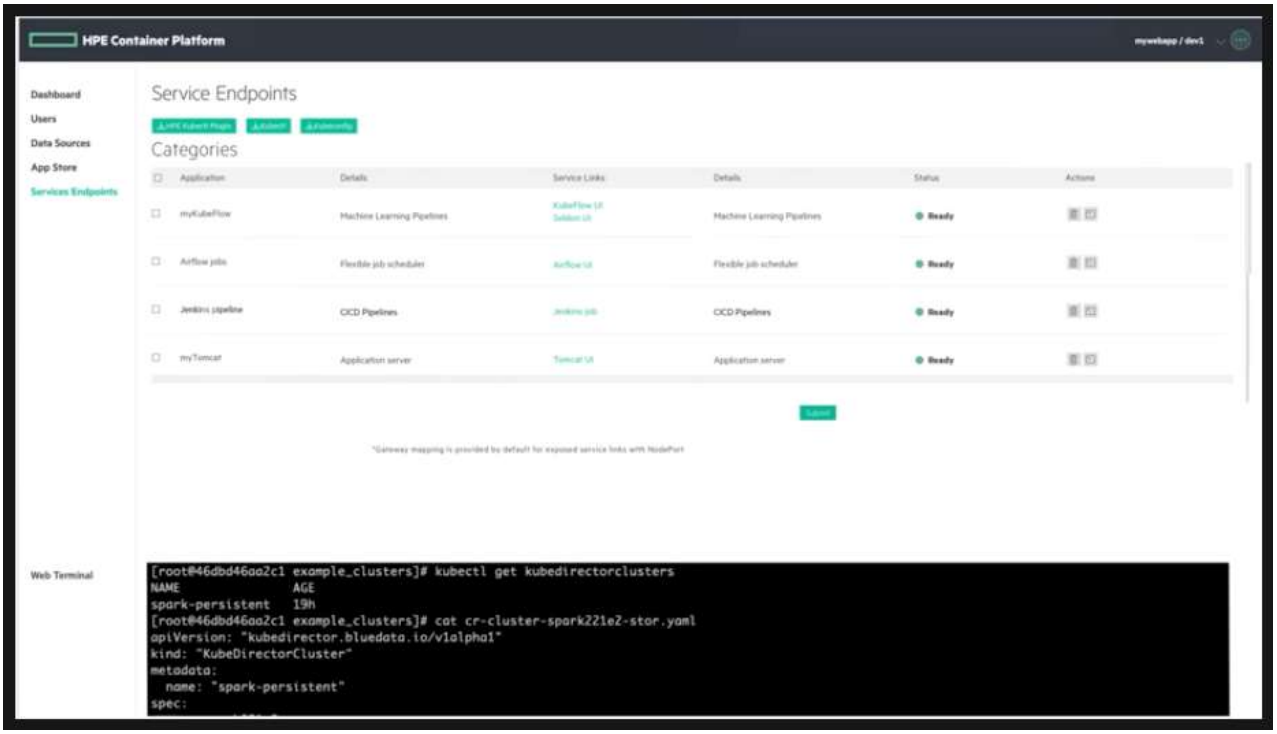
kube-controller-manager

Component on the master that runs controllers.

Logically, each controller is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

These controllers include:

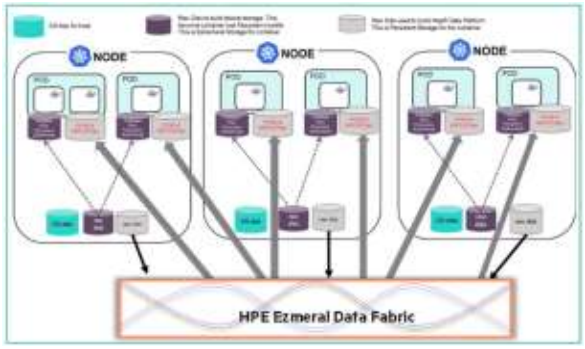
- Node Controller: Responsible for noticing and responding when nodes go down.
- Replication Controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.

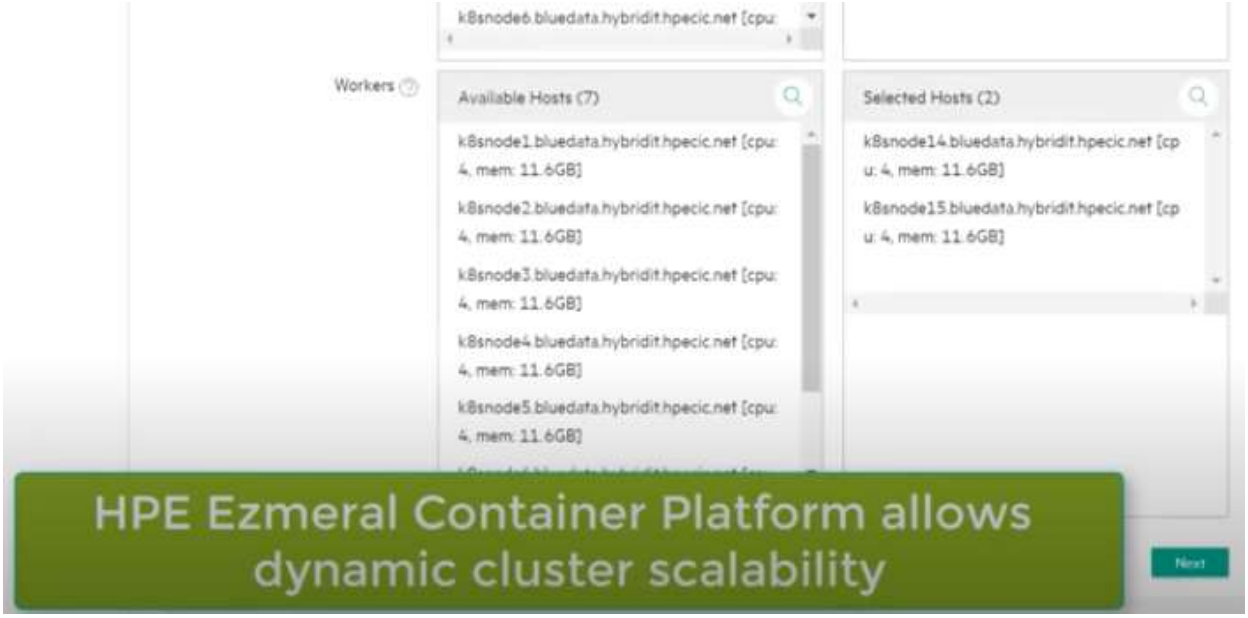


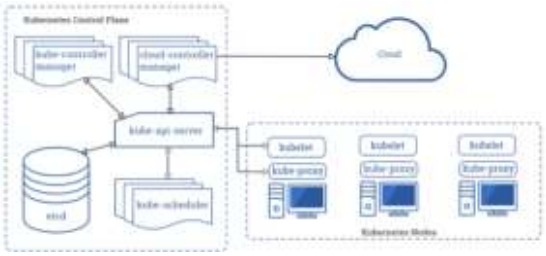
The HPE Ezmeral Container Platform provides an enterprise-class solution to deploy and manage a containerized environment at scale. Developers have secure on-demand access to their environment and can develop apps and release code faster. They can also build once and deploy anywhere with the portability of containers. IT teams can manage multiple Kubernetes clusters with multitenant container isolation and data access for any workload from edge to core to cloud. Moreover, they can extend the benefits of containers beyond the cloud-native microservices-architected stateless applications to containerized monolithic stateful applications with persistent data.

- **Multi-cluster Kubernetes management:** Fast, easy deployment and management, and monitoring of multiple clusters with the out-of-the-box configuration of networking, load balancing, and storage.

72. The Accused '153 Products, when deployed on distributed processing systems of, for example, HPE or its customers, comprise a plurality of network-connected distributed client systems, the client systems having under-utilized capabilities. For example, the HPE Ezmeral Container Platform allows dynamic cluster scalability, identifying available client systems on which distributed project workloads, such as containerized applications, can be run. The multiple Kubernetes nodes that make up an HPE-Ezmeral-configured Kubernetes cluster collectively form a plurality of network-connected distributed client systems having under-utilized capabilities:

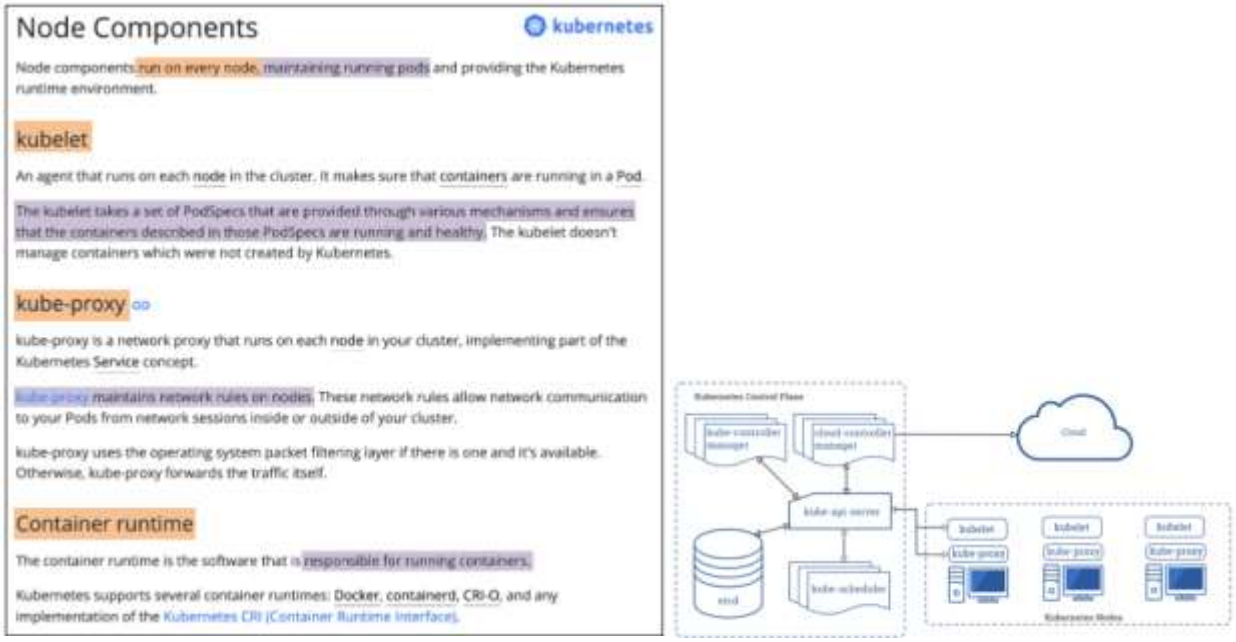




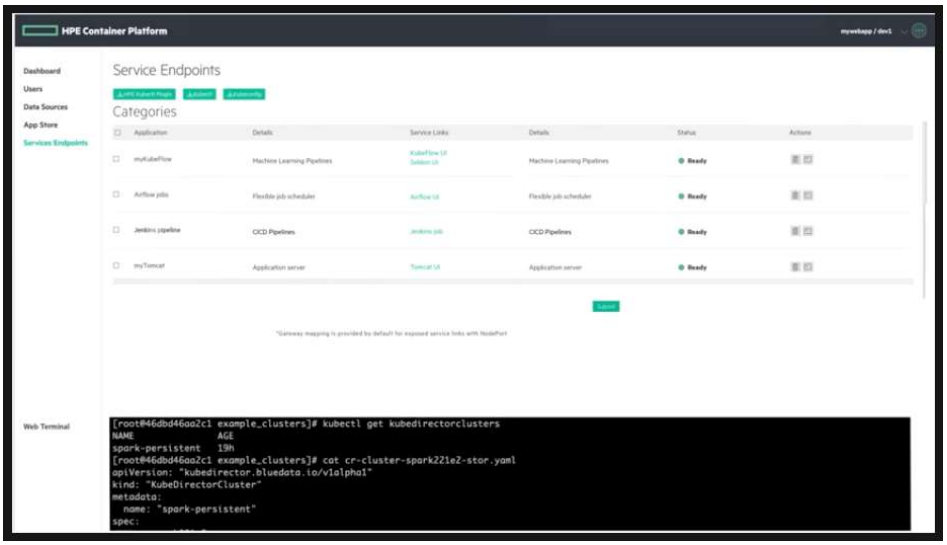


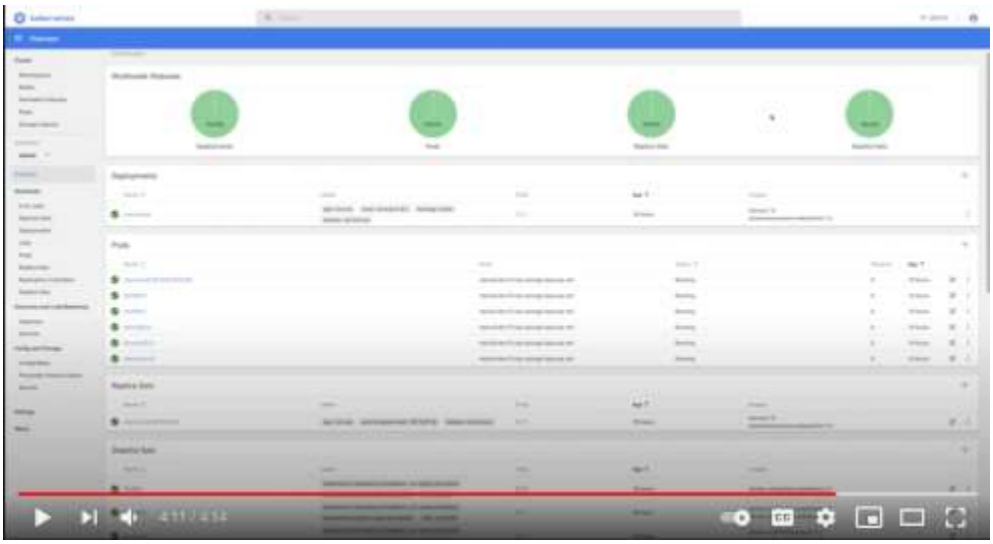
73. The Accused ’153 Products comprise a client agent program configured to run on the client systems and to provide workload processing for at least one project of a distributed computing platform. For example, as depicted below, Kubernetes, which is fully integrated into the Accused Products along with innovations like KubeDirector, provides a client agent program configured to run on the client systems in an HPE Ezmeral Container Platform cluster, including kube-proxy, container runtime, and kubelet Kubernetes components, with the nodes / client

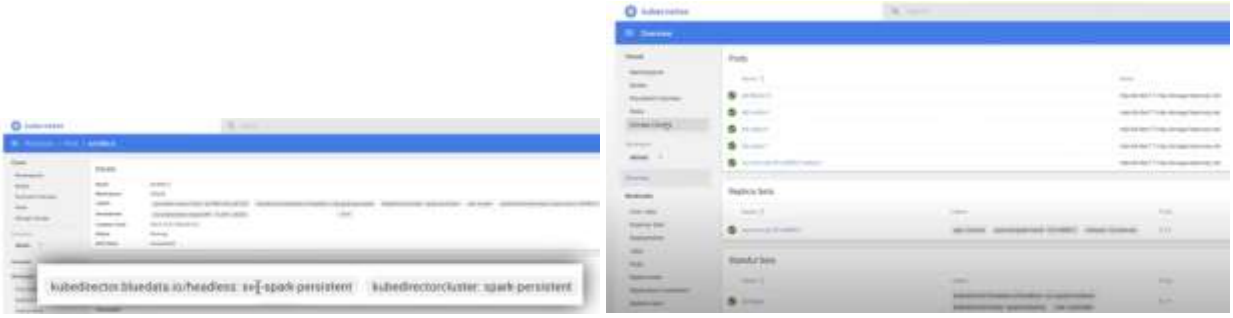
systems of the cluster(s) providing workload processing for a project running on the distributed computing platform (*e.g.*, the HPE Ezmeral Container Platform cluster):

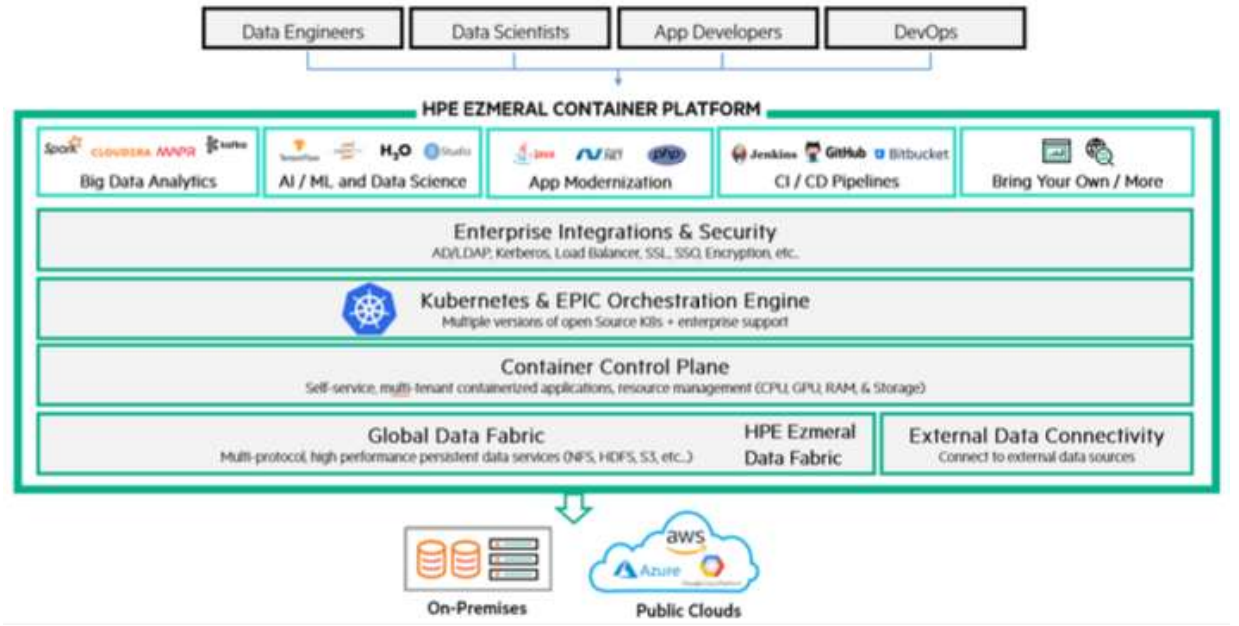


74. The Accused '153 Products comprise at least one server system configured to communicate with the plurality of client systems through a network. As one non-limiting example, HPE Ezmeral Container Platform integrated with a Kubernetes controller manages a Kubernetes cluster comprising client systems running instances of distributed project workloads, such as Apache Spark, and communicates with them through a network:



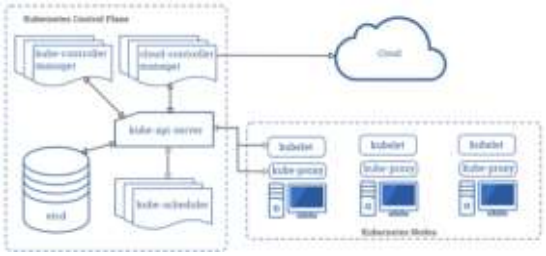






- **Multi-cluster Kubernetes management:** Fast, easy deployment and management, and monitoring of multiple clusters with the out-of-the-box configuration of networking, load balancing, and storage.

- **100% open-source Cloud Native Computing Foundation (CNCF) Kubernetes:** With innovations such as KubeDirector—an open-source Kubernetes-based controller to deploy non-cloud-native, stateful apps. HPE Ezmeral Container Platform is a CNCF certified Kubernetes distribution.



kube-controller-manager

Component on the master that runs controllers.

Logically, each controller is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

These controllers include:

- Node Controller: Responsible for noticing and responding when nodes go down.
- Replication Controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.

75. The Accused '153 Products comprise at least one server system to provide the client agent program to the client systems and to send initial project and poll parameters to the client systems. As one non-limiting example, HPE Ezmeral Container Platform, as integrated with a Kubernetes controller, provides containerized application instances along with kube-proxy, container runtime, and kubelet Kubernetes components to nodes of a Kubernetes cluster, as well as initial parameters relating to a project (*e.g.*, minimum and maximum number of nodes, constraints on node scaling up or down, and requisite number of pods being run to adequately handle workload processing), and initial poll parameters such as rules relating to the specific metrics to be evaluated, evaluation intervals, actions taken when rules are met, and intervals for kubelets to update NodeStatus:

The following parameters are used in an auto-scale policy. See Sample JSON Auto-Scale Policies for three sample policies.

PROPERTY	DESCRIPTION	TYPE	REQUIRED
min_node_count	Minimum number of virtual nodes that can exist in the cluster.	integer	true
max_node_count	Maximum number of virtual nodes that can exist in the cluster.	integer	true
metric_field	Metric item that will be evaluated.	string; will be one of the following: • cpu • memory • rss_total • rss_pct • diskio_write • diskio_read • network_in • network_out	true
greater_than	Define measurement threshold and return true if the value exceeds the threshold. Each condition should include only one greater_than or one less_than property, and these properties are exclusive.	integer	true
less_than	Define measurement threshold and return true if the value is less than the threshold. Each condition should include only one greater_than or one less_than property, and these properties are exclusive.	integer	true
unit	Type of unit to use when measuring the metric item.	string; will be one of the following: • percent • mb • MBPerSec	
action	Action that will occur when the rule is met.	string; will be either: • expand • shrink	true
step			
min_node_count	Number of virtual nodes to add per action when expanding or remove when shrinking the virtual cluster.	integer	true
name	Name of the rule.	string	false
description	Detailed description for the rule.	string	false
evaluation_period	Interval to use for evaluation. When evaluating the conditions, this value specifies how long the historical data should be used. One condition is considered to be true only when the condition is met during the whole evaluation_period.	integer	true
action_interval	Interval between two consecutive scaling actions. Meeting a rule generally triggers the corresponding expand/shrink action; however, some time is needed to distribute the load between virtual nodes, and the pace of scaling the virtual cluster up or down action should be controlled. If the rule is met but the action_period has not yet elapsed, then the rule will be ignored.	integer	true
if_any	The rule is met when any included condition is met. Each rule should only include one if_any condition or one if_all condition set. The if_all condition set and the if_any condition set are exclusive within one rule.	policy condition	false
if_all	The rule is met when all included conditions are met. Each rule should only include one if_any condition or one if_all condition set. The if_all condition set and the if_any condition set are exclusive within one rule.	policy condition	false
role_id	ID of the role object in a catalog entry.	string	true

constraints	Virtual node count limitation when scaling up or down.	policy_constraints	true
rules	User defined rules. The virtual cluster will be scaled up or down when one rule's condition is met.	policy_rule	true
name	User-defined auto-scale policy name.	string	false
version	Version of the auto-scale policy	string	false
policies	User-defined auto-scale policy for each virtual node role.	autoscale_role_policy	true

Heartbeats

Heartbeats, sent by Kubernetes nodes, help determine the availability of a node.

There are two forms of heartbeats: updates of NodeStatus and the Lease object. Each Node has an associated Lease object in the kube-node-lease namespace. Lease is a lightweight resource, which improves the performance of the node heartbeats as the cluster scales.

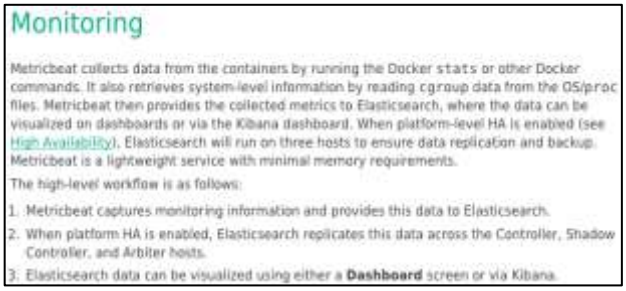
The kubelet is responsible for creating and updating the NodeStatus and a Lease object.

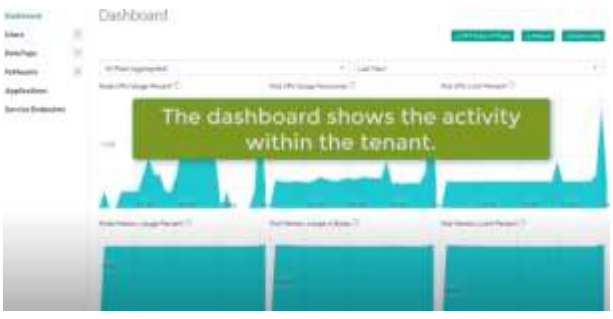
- The kubelet updates the NodeStatus either when there is change in status, or if there has been no update for a configured interval. The default interval for NodeStatus updates is 5 minutes (much longer than the 40 second default timeout for unreachable nodes).

When the job controller sees a new task it makes sure that, somewhere in your cluster, the kubelets on a set of Nodes are running the right number of Pods to get the work done. The job controller does not run any Pods or containers itself. Instead, the job controller tells the API server to create or remove Pods. Other components in the control plane act on the new information (there are new Pods to schedule and run), and eventually the work is done.

After you create a new job, the desired state is for that job to be completed. The job controller makes the current state for that job be nearer to your desired state: creating Pods that do the work you wanted for that job, so that the job is closer to completion.

76. The Accused '153 Products comprise at least one server system to receive poll communications from the client systems during processing of the project workloads, wherein a dynamic snapshot information of current project status is provided based at least in part upon the poll communications from the client systems. For example, the HPE Ezmeral Container Platform receives container- and system-level data through its monitoring and Metricbeat functions; deployment status as part of the KubeDirector reconciliation loop; CPU, memory, and other usage and utilization metrics from Kubernetes clusters (made up of nodes / client systems); and heartbeats sent by Kubernetes nodes to help determine node availability, thus collectively providing dynamic snapshot information of project status. Such dynamic snapshot information can be visualized in multiple ways, including, *e.g.*, on the Kubernetes dashboard accessible directly from the HPE Ezmeral Container Platform's GUI, or on the HPE Ezmeral Container Platform's dashboard itself:





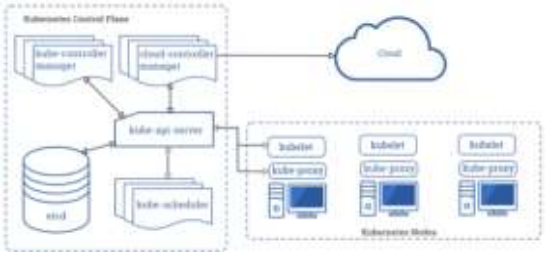
The following two scripts allow Platform Administrators to download Kubernetes usage details:

- **k8susage.py:** Collects the usage metrics from Kubernetes clusters and stores the results in a comma-delimited (.csv) file. Historical usages are collected for the specified time period and aggregated over the specified time interval (aggregation interval) or the current usage (over a 2 minute interval) is collected. Individual pod metrics and pod counts are summed up for each aggregation interval following the hierarchy Pod > Node > Namespace > All namespaces. This script collects the following metrics:
  - CPU cores limits and requests Memory limits and requests
  - Ephemeral storage capacity and usage
  - Number of running and pending pods
  - Tenant storage usage (now option only)
- **k8scsv.py:** Collect utilization metrics from Kubernetes clusters and stores the results in a comma-delimited (.csv) file. Historical usages are collected for the specified time period and aggregated over the specified time interval (aggregation interval) or the current usage (over a 2-minute interval) is collected. Individual pod metrics, except the CPU and memory limit percentages are summed up for each aggregation interval following the hierarchy Pod > Node > Namespace > All Namespaces. For the CPU and memory limit metrics, the per aggregation interval averages are computed for the Node, Namespace, and All Namespaces. This script collects the following metrics:
  - CPU
    - Pod used nanocores
    - Pod usage as a percentage of the total node CPU
    - Pod usage as a percentage of the defined limit for the pod containers (or total node CPU if unlimited)
  - Memory
    - Pod total memory usage
    - Pod memory usage as a percentage of the total node allocable memory
    - Pod memory usage as a percentage of the defined limit for the pod containers (or total node allocable memory if unlimited)
    - Pod total network received (Rx) and transmitted (Tx) bytes
    - Pod network received (Rx) and transmitted (Tx) bytes per aggregation interval (not for

Heartbeats

Heartbeats, sent by Kubernetes nodes, help determine the availability of a node.

There are two forms of heartbeats: updates of `NodeStatus` and the Lease object. Each Node has an associated Lease object in the `kube-node-lease` namespace. Lease is a lightweight resource, which improves the performance of the node heartbeats as the cluster scales.



kube-controller-manager

Component on the master that runs controllers.

Logically, each controller is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

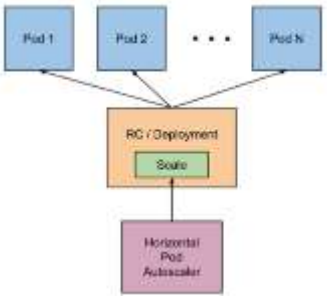
These controllers include:

- Node Controller: Responsible for noticing and responding when nodes go down.
- Replication Controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.

The Horizontal Pod Autoscaler is implemented as a control loop, with a period controlled by the controller manager's `--horizontal-pod-autoscaler-sync-period` flag (with a default value of 15 seconds).

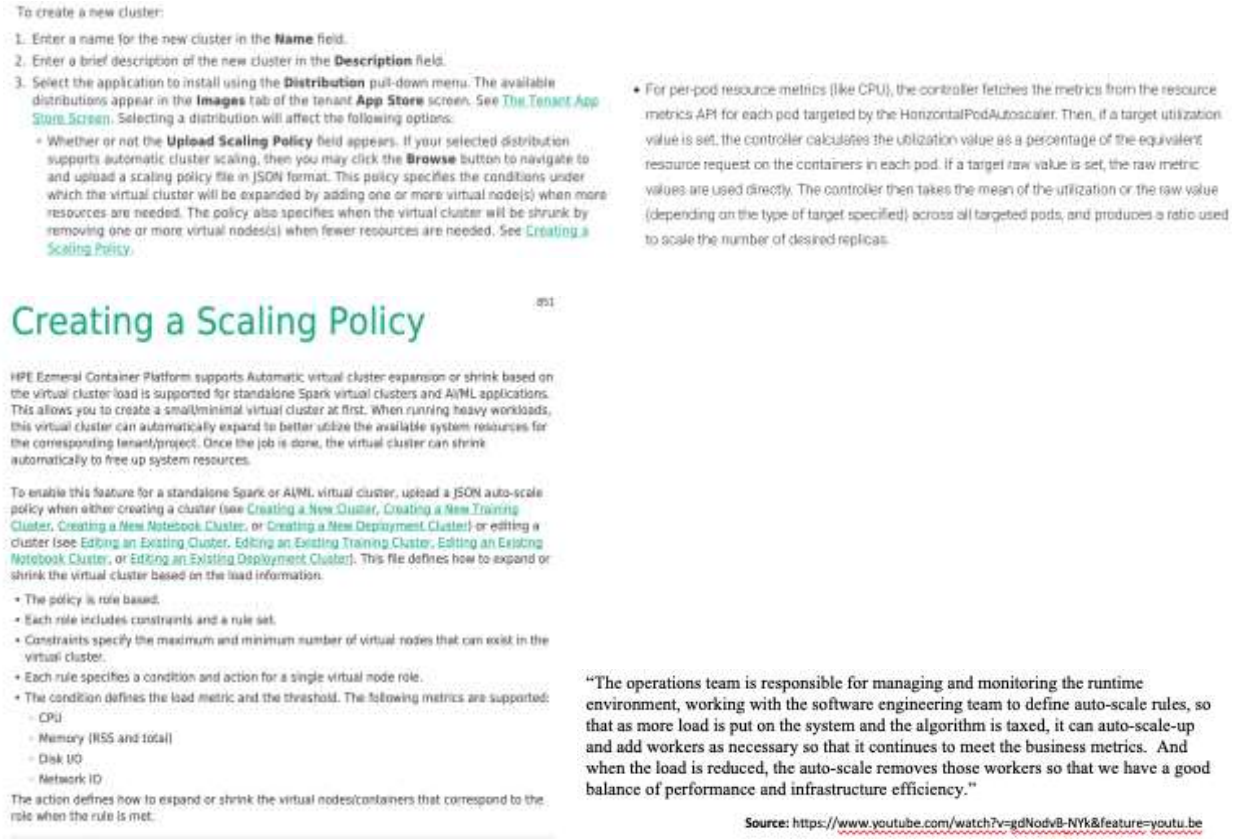
During each period, the controller manager queries the resource utilization against the metrics specified in each HorizontalPodAutoscaler definition. The controller manager obtains the metrics from either the resource metrics API (for per-pod resource metrics), or the custom metrics API (for all other metrics).

- For per-pod resource metrics (like CPU), the controller fetches the metrics from the resource metrics API for each pod targeted by the HorizontalPodAutoscaler. Then, if a target utilization value is set, the controller calculates the utilization value as a percentage of the equivalent resource request on the containers in each pod. If a target raw value is set, the raw metric values are used directly. The controller then takes the mean of the utilization or the raw value (depending on the type of target specified) across all targeted pods, and produces a ratio used to scale the number of desired replicas.



77. The Accused ’153 Products comprise at least one server system to analyze the poll communications utilizing the dynamic snapshot information to determine whether to change how many client systems are active in the at least one project. For example, the HPE Ezmeral Container Platform, as integrated with a Kubernetes controller such as KubeDirector, and with support for choosing or uploading an auto-scaling policy, analyzes metric, heartbeat, load, usage, and utilization information collected from the nodes/client systems within deployed clusters utilizing

dynamic snapshot information about project status to determine the appropriate number of nodes / client systems that are or should be active in the project. For example, fewer client systems (*e.g.*, nodes) are desired, the Accused '153 Products reduce the number of client systems in deployed clusters, and where additional client systems are needed, the Accused Products dynamically add client systems to deployed clusters:



To create a new cluster:

1. Enter a name for the new cluster in the **Name** field.
2. Enter a brief description of the new cluster in the **Description** field.
3. Select the application to install using the **Distribution** pull-down menu. The available distributions appear in the **Images** tab of the tenant **App Store** screen. See The Tenant App Store Screen. Selecting a distribution will affect the following options:
   - Whether or not the **Upload Scaling Policy** field appears. If your selected distribution supports automatic cluster scaling, then you may click the **Browse** button to navigate to and upload a scaling policy file in JSON format. This policy specifies the conditions under which the virtual cluster will be expanded by adding one or more virtual node(s) when more resources are needed. The policy also specifies when the virtual cluster will be shrunk by removing one or more virtual nodes(s) when fewer resources are needed. See Creating a Scaling Policy.

- For per-pod resource metrics (like CPU), the controller fetches the metrics from the resource metrics API for each pod targeted by the HorizontalPodAutoscaler. Then, if a target utilization value is set, the controller calculates the utilization value as a percentage of the equivalent resource request on the containers in each pod. If a target raw value is set, the raw metric values are used directly. The controller then takes the mean of the utilization or the raw value (depending on the type of target specified) across all targeted pods, and produces a ratio used to scale the number of desired replicas.

Creating a Scaling Policy

HPE Ezmeral Container Platform supports Automatic virtual cluster expansion or shrink based on the virtual cluster load is supported for standalone Spark virtual clusters and AI/ML applications. This allows you to create a small/minimal virtual cluster at first. When running heavy workloads, this virtual cluster can automatically expand to better utilize the available system resources for the corresponding tenant/project. Once the job is done, the virtual cluster can shrink automatically to free up system resources.

To enable this feature for a standalone Spark or AI/ML virtual cluster, upload a JSON auto-scale policy when either creating a cluster (see Creating a New Cluster, Creating a New Training Cluster, Creating a New Notebook Cluster, or Creating a New Deployment Cluster) or editing a cluster (see Editing an Existing Cluster, Editing an Existing Training Cluster, Editing an Existing Notebook Cluster, or Editing an Existing Deployment Cluster). This file defines how to expand or shrink the virtual cluster based on the load information.

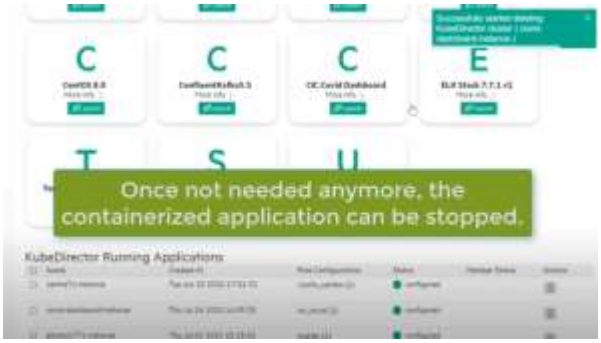
- The policy is role based.
- Each role includes constraints and a rule set.
- Constraints specify the maximum and minimum number of virtual nodes that can exist in the virtual cluster.
- Each rule specifies a condition and action for a single virtual node role.
- The condition defines the load metric and the threshold. The following metrics are supported:
  - CPU
  - Memory (RSS and total)
  - Disk I/O
  - Network IO

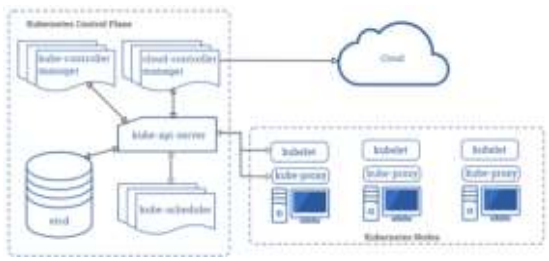
The action defines how to expand or shrink the virtual nodes/containers that correspond to the role when the rule is met.

"The operations team is responsible for managing and monitoring the runtime environment, working with the software engineering team to define auto-scale rules, so that as more load is put on the system and the algorithm is taxed, it can auto-scale-up and add workers as necessary so that it continues to meet the business metrics. And when the load is reduced, the auto-scale removes those workers so that we have a good balance of performance and infrastructure efficiency."

Source: https://www.youtube.com/watch?v=gdNodvB-NYk&feature=youtu.be

Note
This architecture can scale between three (3) and n worker nodes. Also, the HPE Synergy D3940 Storage Module can scale between two (6) and 40 disks per frame.





kube-controller-manager

Component on the master that runs controllers.

Logically, each controller is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

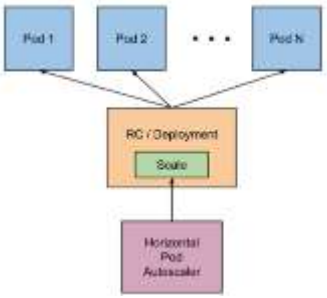
These controllers include:

- Node Controller: Responsible for noticing and responding when nodes go down.
- Replication Controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.

The Horizontal Pod Autoscaler is implemented as a control loop, with a period controlled by the controller manager's `--horizontal-pod-autoscaler-sync-period` flag (with a default value of 15 seconds).

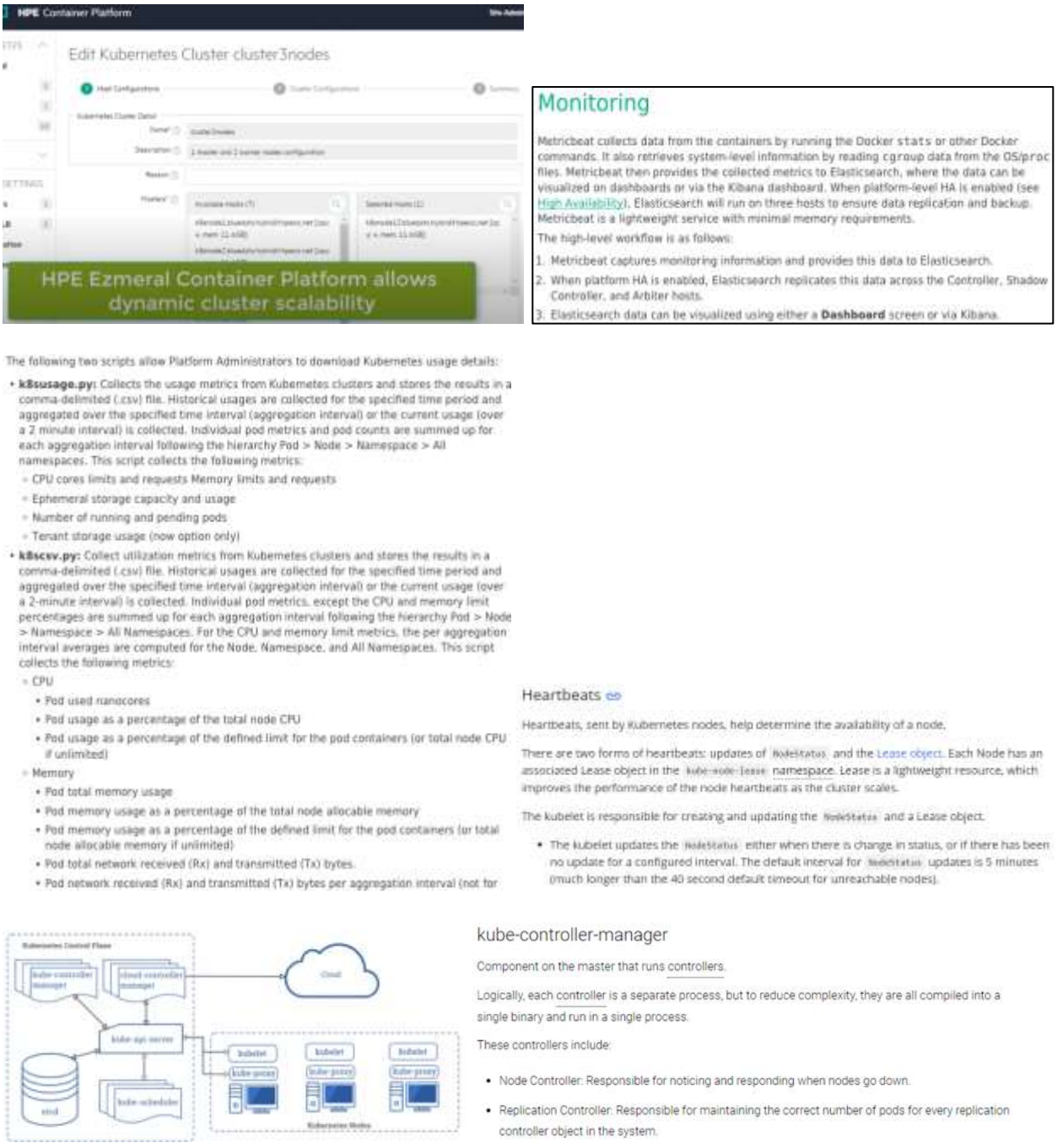
During each period, the controller manager queries the resource utilization against the metrics specified in each HorizontalPodAutoscaler definition. The controller manager obtains the metrics from either the resource metrics API (for per-pod resource metrics), or the custom metrics API (for all other metrics).

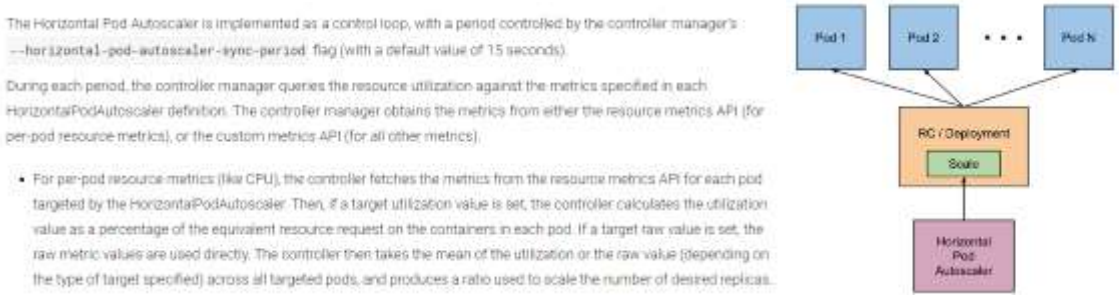
- For per-pod resource metrics (like CPU), the controller fetches the metrics from the resource metrics API for each pod targeted by the HorizontalPodAutoscaler. Then, if a target utilization value is set, the controller calculates the utilization value as a percentage of the equivalent resource request on the containers in each pod. If a target raw value is set, the raw metric values are used directly. The controller then takes the mean of the utilization or the raw value (depending on the type of target specified) across all targeted pods, and produces a ratio used to scale the number of desired replicas.



78. The Accused '153 Products comprise at least one server system to repeatedly utilize the poll communication and the poll response communications to coordinate project activities of the client systems during project operations. For example, the HPE Ezmeral Container Platform,

as integrated with a Kubernetes controller such as KubeDirector, and with support for choosing or uploading an auto-scaling policy, coordinates project activities dynamically, on an ongoing basis, receiving, *e.g.*, metric data and load information at regular or defined intervals and scaling up or down the number of client systems working on the project as appropriate:





79. Additionally, HPE has been, and currently is an active inducer of infringement of the ’153 patent under 35 U.S.C. § 271(b) and a contributory infringer of the ’153 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

80. HPE has actively induced, and continues to actively induce, infringement of the ’153 patent by intending that others use, offer for sale, or sell in the United States products and/or services covered by the ’153 patent, including but not limited to the Accused ’153 Products and any other HPE product and/or service, alone or in combination, that operates in materially the same manner. For example, HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, use, in accordance with HPE’s design, intent and directions, provision for use, offer for sale, or sell in the United States products and/or services that directly infringe the ’153 patent.

81. HPE has contributed to, and continues to contribute to, the infringement of the ’153 patent by others by offering to sell, selling, or otherwise commercially offering products and/or services that, when installed and configured result in a system as intended by HPE, that directly infringes the ’153 patent.

82. HPE knew of the ’153 patent, or should have known of the ’153 patent, but was willfully blind to its existence. HPE has had actual knowledge of the ’153 patent since at least as early as service upon HPE of this Complaint. Upon information and belief, HPE has had actual knowledge of the ’153 patent since at least as early as the service upon HPE of this Complaint.

On information and belief, HPE had actual knowledge of the ’153 patent in and around August 18, 2020. On information and belief, HPE was aware of IV’s allegations against Arista Networks, Inc. relating to infringement of the ’153 patent through the use of Kubernetes on nodes and servers in a distributed computing platform with dynamic coordination capabilities. Additionally, HPE was aware of the ’153 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the ’153 patent.

83. HPE has committed, and continues to commit, affirmative acts that cause infringement of the ’153 patent with knowledge of the ’153 patent and knowledge or willful blindness that the induced acts constitute infringement of the ’153 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customer way as desired and intended by HPE, infringe the ’153 patent and/or by directly or indirectly providing instruction on how to use its products and/or services in a manner or configuration that infringes the ’153 patent, including those found at the following:

- https://assets.ext.hpe.com/is/content/hpedam/a50002683enw
- https://support.hpe.com/hpesc/public/docDisplay?docLocale=en_US&docId=a00111243en_us
- https://assets.ext.hpe.com/is/content/hpedam/a50002075enw
- https://h20195.www2.hpe.com/V2/GetPDF.aspx/a00039700enw.pdf
- https://www.youtube.com/watch?v=OtJh6Mbouks
- https://www.youtube.com/watch?v=OLuQC2yAsCM

- https://www.youtube.com/watch?v=OtJh6Mbouks&t=95s
- https://www.hpe.com/us/en/greenlake/container-platform.html
- https://docs.containerplatform.hpe.com/51/51/kubernetes-and-hpe-greenlake-f/general-functionality--include/Getting_Started_with_Kubernetes_in_HPE_GreenLake.html
- https://support.hpe.com/hpesc/public/docDisplay?docId=a00092451en_us&page=GUID-156B9779-456B-4D31-AFE0-755434D4AC12.html
- https://support.hpe.com/hpesc/public/docDisplay?docId=a00092451en_us&page=GUID-E7C954DF-322E-44D9-8072-61F176E0BB7C.html

84. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that, when used, cause the direct infringement of the ’153 Patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such as software, that are especially made or especially adapted for use in infringement of the ’153 patent that are not a staple article or commodity of commerce suitable for substantial non-infringing use.

85. As a result of HPE’s acts of infringement, IV has suffered and will continue to suffer damages in an amount to be paid at trial.

COUNT III
(HPE’s Infringement of U.S. Patent No. 7,783,788)

86. Paragraphs 1-85 are incorporated by reference as if fully set forth herein.

87. The inventions claimed by the ’788 patent, taken alone or in combination, were not well-understood, routine or conventional to one of ordinary skill in the art at the time of the invention of the’788 patent, around 2006. Rather, the patent teaches and claims an improved way to virtualize input/output (“I/O”) subsystems that facilitates transparent sharing of I/O subsystems among multiple processing systems. For example, the inventions improved upon then-existing virtual I/O systems, which were limited by more static infrastructures, storage, and network

management configurations, by using network fabric, protocol stack logic, and virtualization logic to establish persistent connections between I/O subsystem physical resources and the applications and servers accessing the subsystems and emulating the application servers relative to the subsystem's resources. Importantly, the invention also included controlling utilization of the I/O subsystems by different servers by controlling a physical interface used to access the storage resources. Specifically, the invention requires I/O subsystem device protocol stack logic operative to control data transfer with one or more peripheral systems, establishing a persistent control connection to the virtual I/O peripheral subsystem module of an application server, transmitting I/O peripheral subsystem configurations to the server, emulating the application server relative to the peripheral system, intermediating the I/O subsystem traffic between the application server and peripheral system, and controlling the utilization of subsystem resources by the application servers via the I/O subsystem physical interface per a configured allocation.

88. This is to be contrasted with then-existing systems that, for example, did not allow for peripheral subsystems, such as network and storage resources, to be entirely virtualized and transparently shared with application servers, and did not enforce granular and configurable resource allocations with respect to application server usage across physical interfaces leading to those virtualized resources as the virtualized peripheral subsystems engaged in I/O operations. This resulted in an inefficient allocation of network resources to such virtual servers, or an uncontrolled and unregulated allocation of resources to the servers, which in turn lead to suboptimal I/O system performance. Application servers, for example, were not previously able to obtain tailored I/O peripheral subsystem logical resource allocations, that are today required to handle the wide variety of communications typical of I/O systems in a virtualized environment.

One example of such typical I/O communications are high bandwidth distributed storage read/write operations to/from a multiplicity of servers.

89. The inventions represented a technical solution to an unsolved technological problem. The specification of the '788 patent describes, in technical detail, each of the limitations in the claims, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claims differ markedly from what had been conventional or generic in the industry at the time of the invention of the '788 patent. More specifically, the claims of the '788 patent require a memory, one or more processors, an input/output fabric interface and an I/O subsystem physical interface. Further, the claims require I/O subsystem device protocol stack logic operative to control data transfer with one or more peripheral systems over the I/O subsystem physical interface, virtualization logic for; establishing a persistent control connection to the virtual I/O peripheral subsystem module of an application server, transmitting I/O peripheral subsystem configurations to the server, emulating the application server relative to the peripheral system, intermediating the I/O subsystem traffic between the application server and peripheral system, and controlling the utilization of resources by the application servers per a configured allocation.

90. The systems and methods covered by the asserted claims therefore differ markedly from the conventional and generic systems in use at the time of this invention, which, *inter alia*, lacked the ability to provide I/O peripheral subsystems as logical abstractions to be transparently shared among application servers, which are emulated relative to the peripheral systems, and to control utilization of the I/O peripheral subsystems by allocating I/O subsystem physical interface resources. In turn, such functionality greatly increases the efficiency of the system as a whole and

eliminates over or under provisioning and provides for resource allocation decisions that meet more stringent QoS (Quality of Service) requirements.

91. As described above, the ’788 patent is drawn to solving a specific, technical problem arising in the context of facilitating and managing I/O subsystem operations in a virtualized environment. Consistent with the problem addressed being rooted in such I/O subsystem management techniques, the solutions provided by the ’788 patent naturally are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

92. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 1 of the ’788 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the ’788 patent. HPE’s products and/or services that infringe the ’788 patent include, but are not limited to, the HPE 3PAR StoreServ family of flash-optimized data storage systems, the HPE Primera Storage, HPE GreenLake Enterprise-Ready VM Service when implemented with Primera Storage platform, HPE GreenLake Mission Critical Storage Service when implemented with Primera Storage platform, and any other of HPE’s products and/or services, either alone or in combination, that operate in substantially the same manner (together the “Accused ’788 Products”).

Claim 1 of the ‘788 patent is reproduced below:

> *1. An apparatus, comprising*
> *a memory;*
> *one or more processors;*
> *an input/output (I/O) fabric interface;*
> *an I/O subsystem physical interface;*
> *I/O subsystem device protocol stack logic operative to control data transfer with one or more peripheral systems over the I/O subsystem physical interface; and*
> *virtualization logic encoded in one or more tangible media for execution and when executed operable to cause the one or more processors to:*

establish one or more persistent control connections to virtual I/O peripheral subsystem interface driver modules of one or more application servers;

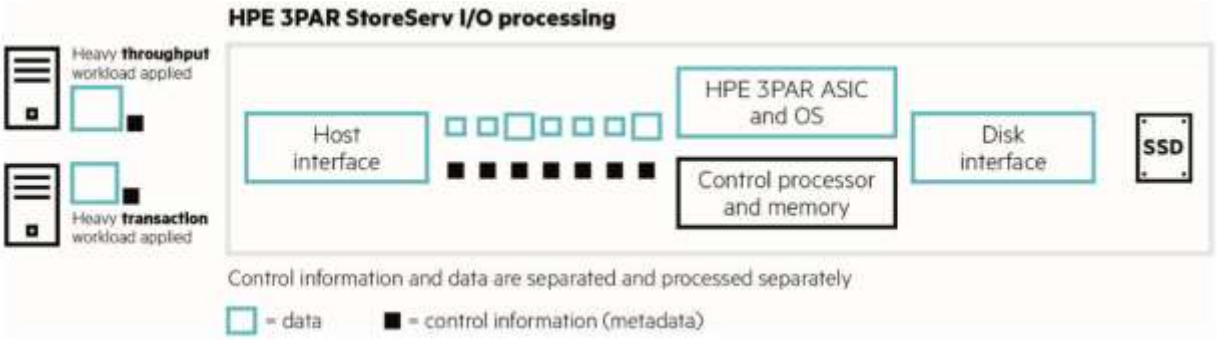
transmit I/O peripheral subsystem configurations to the one or more application servers over the respective one or more persistent control connections;

emulate, relative to the one or more peripheral systems, the one or more application servers;

intermediate I/O subsystem traffic between the one or more application servers and the one or more peripheral systems; and

control utilization of resources of the I/O subsystem physical interface by the one or more application servers according to a configured allocation of resources for the I/O subsystem physical interface across the one or more application servers.

93. As one non-limiting example, the Accused '788 Products include a memory and one or more processors as seen below:



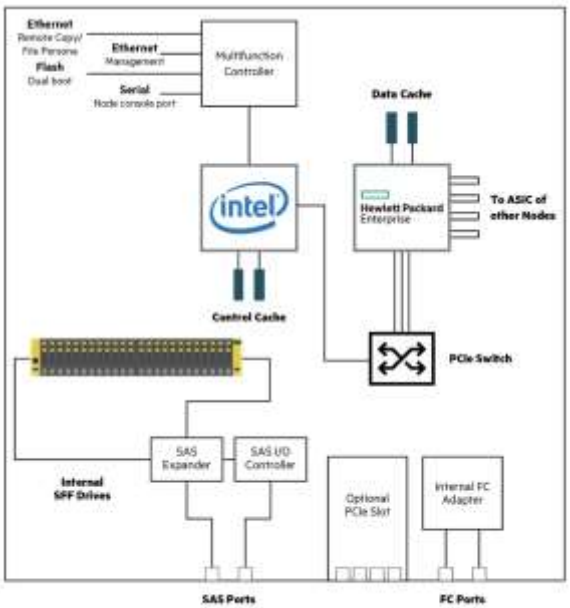


FIGURE 6. HPE 3PAR 8000 Controller Node

94. The Accused '788 Products also include an input/output fabric interface. For instance, the HPE 3PAR system ships with a plurality of ports to a backplane:

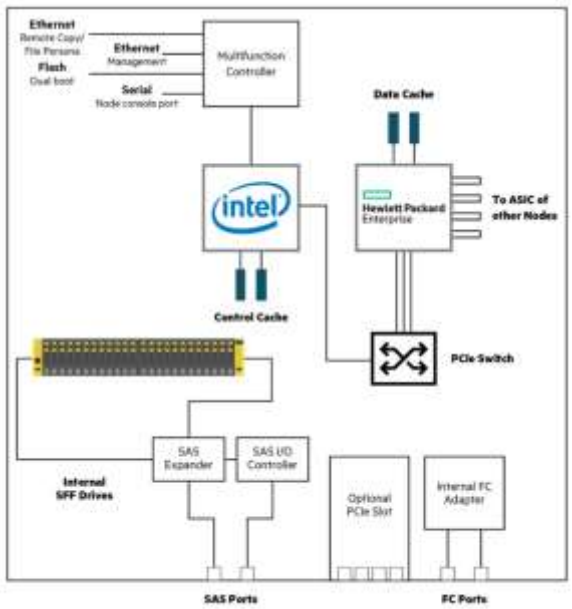
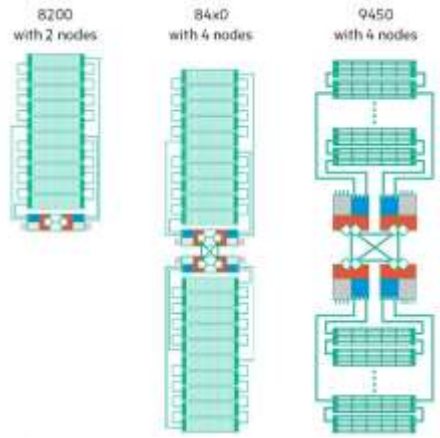


FIGURE 6. HPE 3PAR 8000 Controller Node



95. Furthermore, the Accused '788 Products include an I/O subsystem physical interface. For instance, the HPE 3PAR system includes one or more Fibre Channel, iSCSI, FCoE or Ethernet ports as illustrated below:

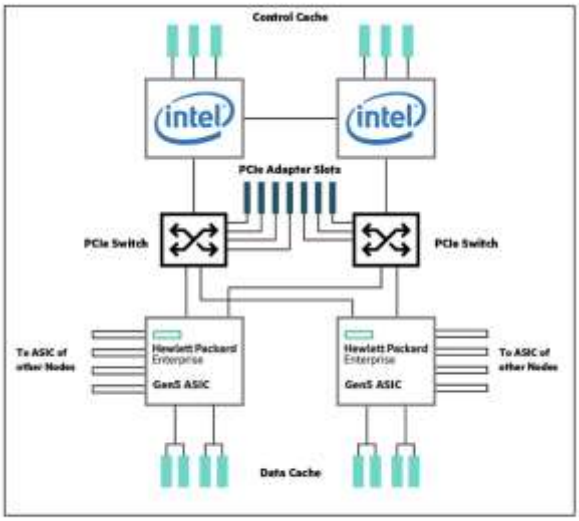


FIGURE 3. Controller node design of the HPE 3PAR 9450 and 20000 Storage system

Each HPE 3PAR 9000 and 20000 series controller node can have a maximum of the following ports per node:

- (10) Dual-port 32 Gbps Fibre Channel adapter
- (20) Quad-port 16 Gbps Fibre Channel adapter
- (10) Dual-port 10 Gbps iSCSI or Fibre Channel over Ethernet (FCoE) converged network adapter
- (6) 10 Gbps Ethernet adapter for file and object access services using File Persona

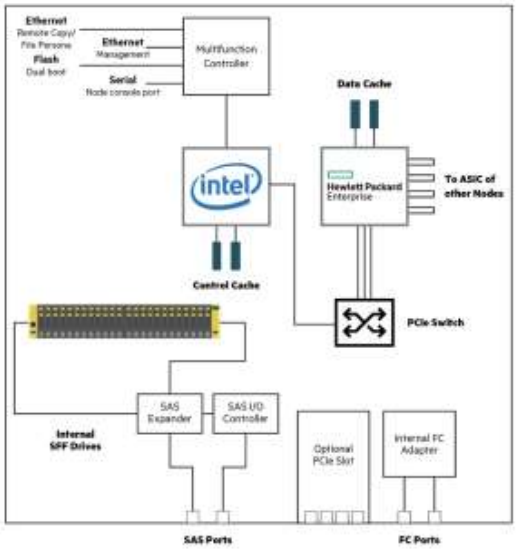


FIGURE 4. HPE 3PAR 8000 Controller Node

Each HPE 3PAR 8000-series controller node has two built-in 4 port 16 Gbps Fibre Channel ports and one PCIe expansion slot. This slot can hold one of the following adapters:

- Dual-port 32 Gbps Fibre Channel adapter
- Quad-port 16 Gbps Fibre Channel adapter
- Dual-port 10 Gbps iSCSI or Fibre Channel over Ethernet (FCoE) converged network adapter
- Quad-port 16 Gb Fibre Channel/10 Gb NIC Combo Adapter
- Quad-port 10 Gb iSCSI/10 Gb NIC Combo Adapter
- Dual-port 10 Gbps Ethernet adapter for file and object access services using HPE 3PAR File Persona
- Quad-port 1 Gbps Ethernet adapter for file and object access services using HPE 3PAR File Persona

96. The Accused '788 Products also include I/O subsystem device protocol stack logic operative to control data transfer with one or more peripheral systems over the physical interface. For example, the HPE 3PAR system includes Priority Optimization software that implements and manages priority policies per virtual volume set and virtual domain. The aforementioned software operates on I/O communications between the HPE 3PAR system and a host:

HPE 3PAR Priority Optimization software for HPE 3PAR StoreServ Storage systems implements and manages a priority policy per virtual volume set (VVset) that serves as a proxy for applications and per virtual domains that serves as a proxy for customers or tenants. Volumes that are not a member of a VVset or a virtual domain can be controlled using the System QoS rule. Priority Optimization enables users to take full control of performance by specifying minimum and maximum limits ("Min Goal" and "Max Limit") for IOPS and bandwidth, as well as a priority for every QoS object along with the ability to define latency goals for the most important applications. If these goals are not met, the system automatically adjusts the service levels of lower-priority applications and workloads in order to make sure that necessary QoS levels for the highest priority applications are always maintained. This paradigm is valid on HPE 3PAR StoreServ systems because all

Mode of operation

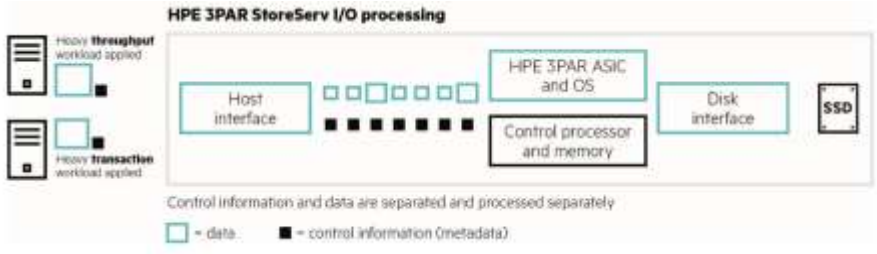
Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

Each HPE 3PAR Storage system features a high-speed, full-mesh passive interconnect that joins multiple controller nodes (the high-performance data movement engines of the HPE 3PAR Architecture) to form a Mesh-Active cluster. This low-latency interconnect allows for tight coordination among the controller nodes to create a system-wide cache which is global, coherent and fault tolerant.

In every HPE 3PAR Storage system, each controller node has a dedicated link to each of the other nodes that operates at 4 GiB/s in each direction. In an HPE 3PAR 20800 Storage system, a total of 56 of these links form the array's full-mesh backplane. In addition, each controller node may have one or more paths to hosts—either directly or over a storage area network (SAN). The clustering of controller nodes enables the system to present hosts with a single, highly available, high-performance storage system. This means that servers can access volumes over any host-connected port—even if the physical storage for the data is connected to a different controller node. This is achieved through an extremely low-latency data transfer across the high-speed, full-mesh backplane.

Priority Optimization supports nesting QoS rules. This allows for the creation of secondary QoS rules on individual VVsets or groups of them in the domain of the tenant. These nested rules limit I/O for smaller objects within a tenant's domain. Nested QoS rules can be overprovisioned without the risk of adversely affecting the workloads of other tenants because the top-level QoS rule enforces a Max Limit on the I/O capacity of all objects within the tenant's domain.



Drive enclosures

Another key element of the HPE 3PAR Storage system is the drive enclosure or drive chassis, which serves as the capacity building block within the system. This section looks in more detail at the different drive enclosures supported by HPE 3PAR Storage systems.

TABLE 3. Capacity building block for HPE 3PAR 20000 systems

	HPE 3PAR 20800	HPE 3PAR 20840	HPE 3PAR 20850	HPE 3PAR 20450
Controller Nodes	2, 4, 6, 8	2, 4, 6, 8	2, 4, 6, 8	2, 4
Max capacity (raw)	9600 TiB	9600 TiB	9043 TiB	4031 TiB
Max usable file capacity	1024 TiB	1024 TiB	512 TiB	256 TiB
Hard drives	8–2304	2304	Not applicable	Not applicable
SSDs	6–1152	6–1152	6–1152	6–576

TABLE 3. Capacity building block for HPE 3PAR 8000 systems

	HPE 3PAR 8450	HPE 3PAR 8440	HPE 3PAR 8400	HPE 3PAR 8200
Controller Nodes	2, 4	2, 4	2, 4	2
Max capacity (raw)	2351 TiB	4000 TiB	2400 TiB	1000 TiB
Max usable file capacity	2–512 TiB	2–512 TiB	2–512 TiB	2–256 TiB
Hard drives	Not applicable	6–960	6–576	6–240
SSDs	6–480	6–480	6–240	6–120

TABLE 4. Capacity building block for HPE 3PAR 9450 systems

	HPE 3PAR 9450
Controller Nodes	2, 4
Max capacity (raw)	6000 TiB
Max usable file capacity	512 TiB
Hard drives	Not applicable
SSDs	6–576

97. The Accused ’788 Products further include virtualization logic. The HPE 3PAR system provides virtual storage for servers, such as application servers, via a three-layer abstraction scheme:

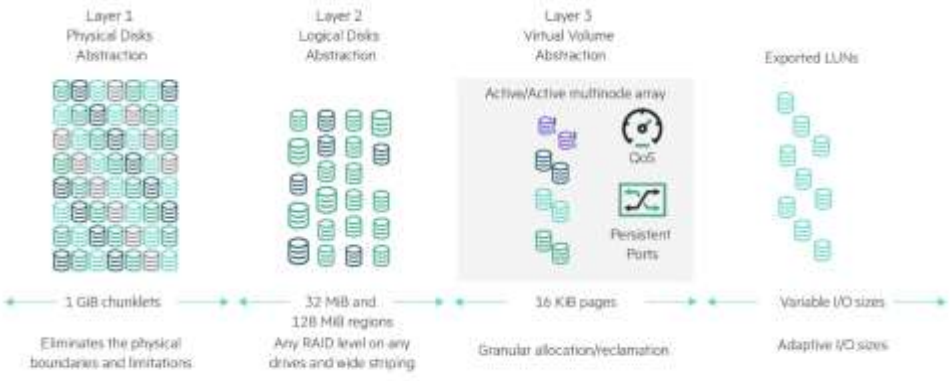
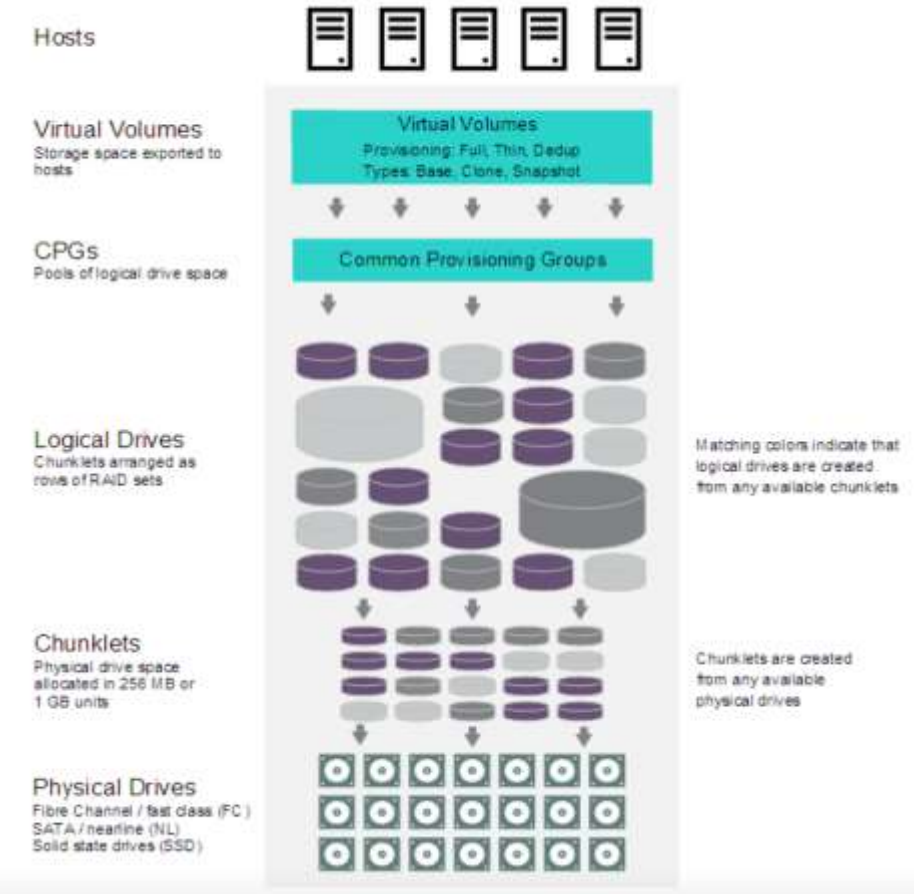


FIGURE 5. Virtualization with a tri-level mapping methodology that provides three layers of abstraction



98. The Accused '788 Products cause the establishing of one or more persistent control connections to virtual I/O peripheral subsystem interface driver modules of one or more application servers. As an example, the 3PAR system's three-layer abstraction scheme and management software abstracts physical drives into chunks, logical disks and virtual volumes before exporting as LUNs via iSCSI or FCoE, as well as allows for converged management of both file and block under a single pane of glass:

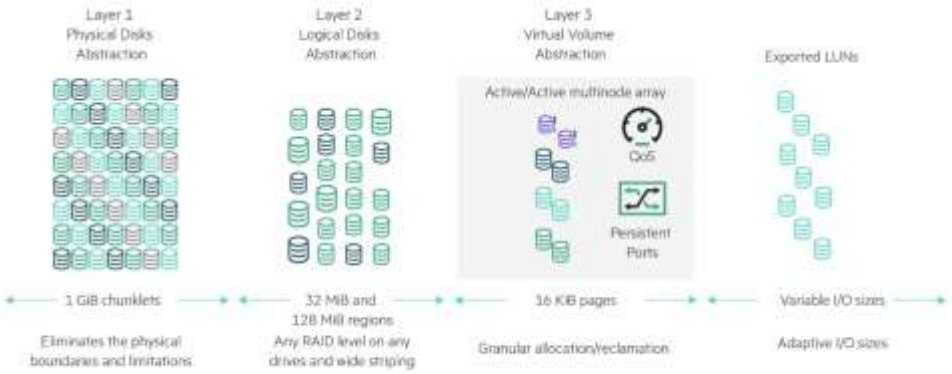
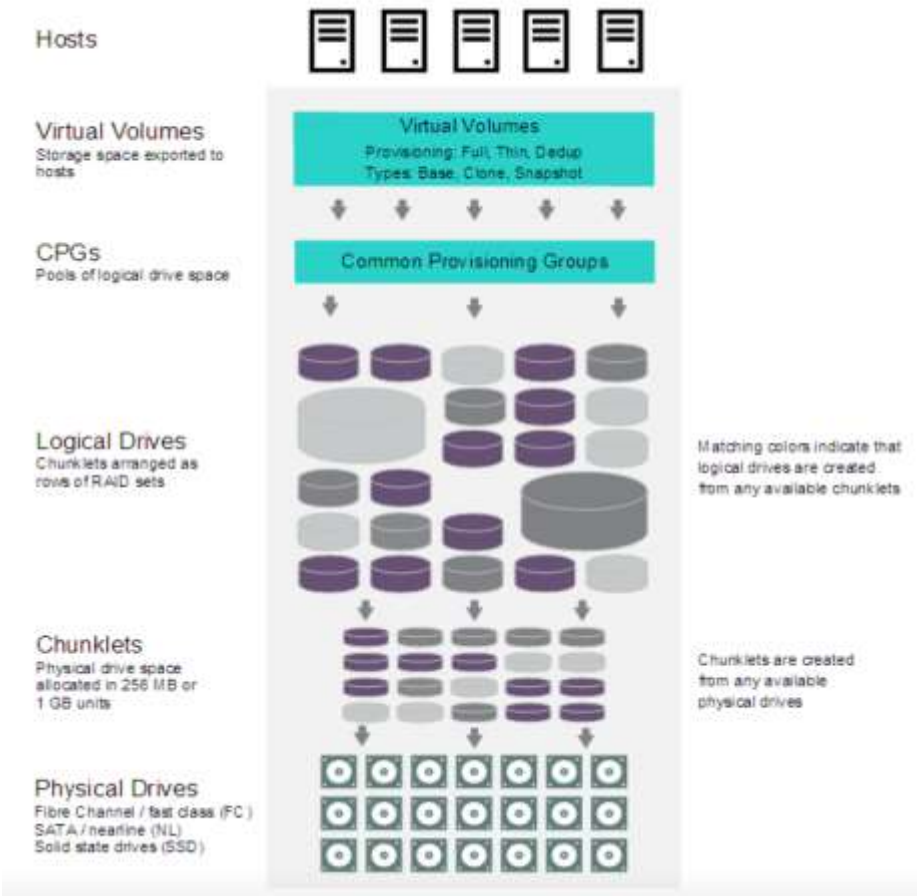


FIGURE 5. Virtualization with a tri-level mapping methodology that provides three layers of abstraction



Virtualization software

Virtualization platforms such as VMware®, Microsoft® Hyper-V®, and Citrix® XenServer® execute business-critical and latency-sensitive applications in parallel with other less-critical applications. It is important to prioritize I/O requests from important applications over the requests from less important applications. The virtualization software creates and manages virtual data stores that are container files comprising one or more virtual drives used by virtual machines (VMs). Each container file is built on one or more storage LUNs that reside on the storage array and are usually accessed over the SAN. Applying a Priority Optimization QoS rule to the LUNs that make up a logical storage container will control the IOPS or bandwidth for all VMs whose virtual drives are contained in that LUN. Be sure that there is enough bandwidth and IOPS in the QoS rule so that the applications running on the VMs can deliver acceptable I/O response times.

Virtual volumes overview

Virtual volumes are created from pools of storage called common provisioning groups and can be exported to hosts. Virtual volumes are the only data layer in HPE 3PAR storage virtualization that is visible to hosts.

LUNs

A LUN is used by hosts to identify exported virtual volumes. The term LUN is a SCSI convention. The terms LUN, disk, drive, and virtual volume are often used interchangeably.

Exported virtual volumes (VLUNs)

Exporting a virtual volume makes it available to hosts by creating an association between the virtual volume on a storage system and a logical unit number (LUN) on a host. The association, called a virtual LUN (VLUN), defines the association between a virtual volume and a host.

A virtual volume can be exported to a host by multiple paths that differ by the host ports on the storage system, host WWNs, host iSCSI names, or host names. Thus, a virtual volume can have multiple VLUNs.

The third layer of abstraction maps LDs to Virtual Volumes (VVs), with all or portions of multiple underlying LDs mapped to the VV. VVs are the virtual capacity representations that are ultimately exported to hosts and applications as virtual LUNs (VLUNs) over Fibre Channel, iSCSI, or FCoE target ports. A single VV can be coherently exported through as few as two ports or as many as ports as desired (no fewer than two, one from each of two different nodes as a minimum). This layer of abstraction uses a table-based association—a mapping table with a granularity of 128 MB per region and an exception table with a granularity of 16 KB per page—as opposed to an algorithmic association. With this approach, a very small portion of a VV associated with a particular LD can be quickly and non-disruptively migrated to a different LD for performance or other policy-based reasons, whereas other architectures require migration of the entire VV. This layer of abstraction also implements many high-level features such as snapshots, caching, pre-fetching, and remote replication.

HPE 3PAR StoreServ Management Console (SSMC) is the HPE 3PAR management and reporting console that offers converged management of both File and Block on HPE 3PAR StoreServ Storage systems. HPE 3PAR StoreServ Management Console offers a modern look and consistent feel and a common interface and language with HPE Management tools such as HPE OneView. Designed to use the latest API (Application Program Interface) and UI (User Interface) technologies, HPE 3PAR SSMC centralizes all HPE 3PAR StoreServ management under a single pane of glass and offers converged management and reporting for both file and block. SSMC 3.5 introduces HPE 3PAR workload Insights that brings in application aware workload management to HPE 3PAR StoreServ Systems.

99. The Accused ’788 Products transmit I/O peripheral subsystem configurations to the one or more application servers over one or more persistent control connections. Following from the above example, the 3PAR system exports virtual volumes as LUNs to an application server over iSCSI or FCoE connections:

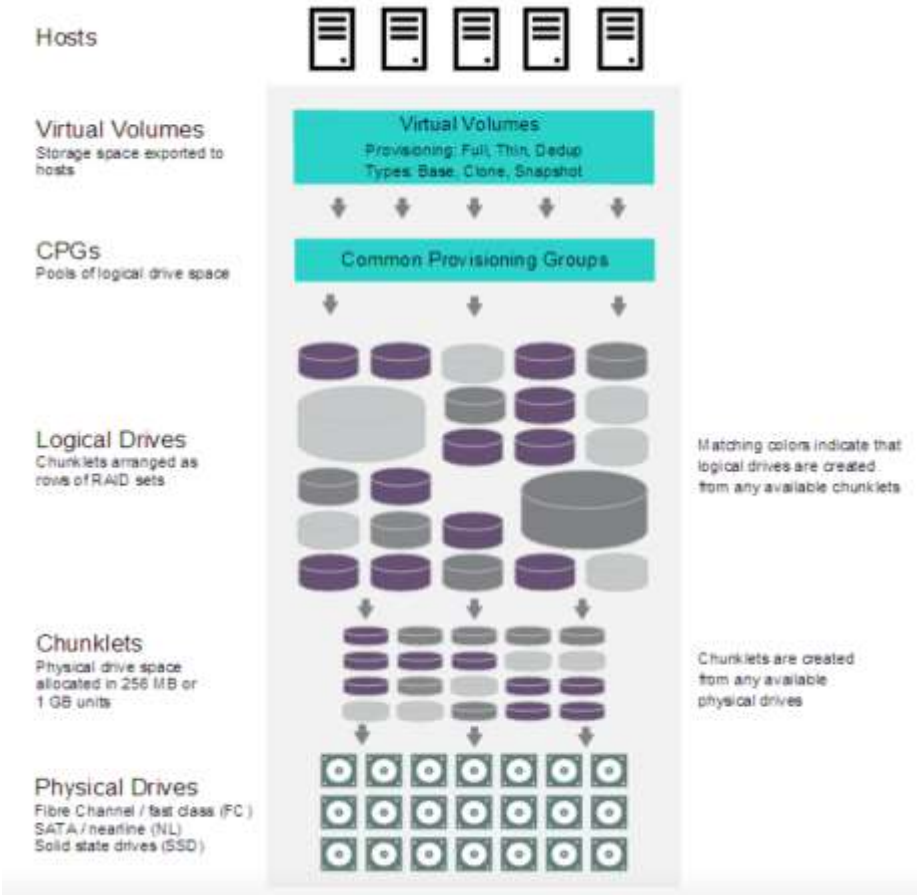


FIGURE 5. Virtualization with a tri-level mapping methodology that provides three layers of abstraction

Exported virtual volumes (VLUNs)

Exporting a virtual volume makes it available to hosts by creating an association between the virtual volume on a storage system and a logical unit number (LUN) on a host. The association, called a virtual LUN (VLUN), defines the association between a virtual volume and a host.

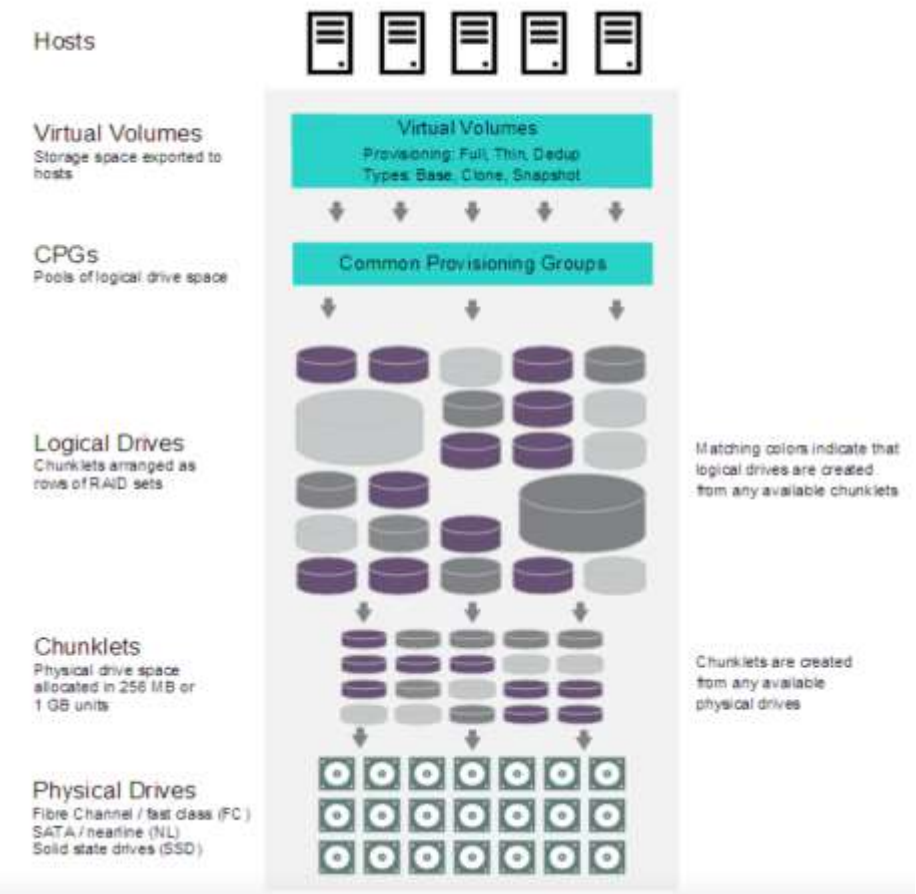
A virtual volume can be exported to a host by multiple paths that differ by the host ports on the storage system, host WWNs, host iSCSI names, or host names. Thus, a virtual volume can have multiple VLUNs.

LUNs

A LUN is used by hosts to identify exported virtual volumes. The term LUN is a SCSI convention. The terms LUN, disk, drive, and virtual volume are often used interchangeably.

The third layer of abstraction maps LDs to Virtual Volumes (VVs), with all or portions of multiple underlying LDs mapped to the VV. VVs are the virtual capacity representations that are ultimately exported to hosts and applications as virtual LUNs (VLUNs) over Fibre Channel, iSCSI, or FCoE target ports. A single VV can be coherently exported through as few as two ports or as many as ports as desired (no fewer than two, one from each of two different nodes as a minimum). This layer of abstraction uses a table-based association—a mapping table with a granularity of 128 MB per region and an exception table with a granularity of 16 KB per page—as opposed to an algorithmic association. With this approach, a very small portion of a VV associated with a particular LD can be quickly and non-disruptively migrated to a different LD for performance or other policy-based reasons, whereas other architectures require migration of the entire VV. This layer of abstraction also implements many high-level features such as snapshots, caching, pre-fetching, and remote replication.

100. The Accused '788 Products emulate, relative to the one or more peripheral systems, the one or more application servers. For example, the 3PAR system's hosts/application servers are presented exported virtual LUNs and appear from the perspective of peripheral system to be directly attached local resources:



Virtual volumes overview

Virtual volumes are created from pools of storage called common provisioning groups and can be exported to hosts. Virtual volumes are the only data layer in HPE 3PAR storage virtualization that is visible to hosts.

Fine-grained approach to virtualization

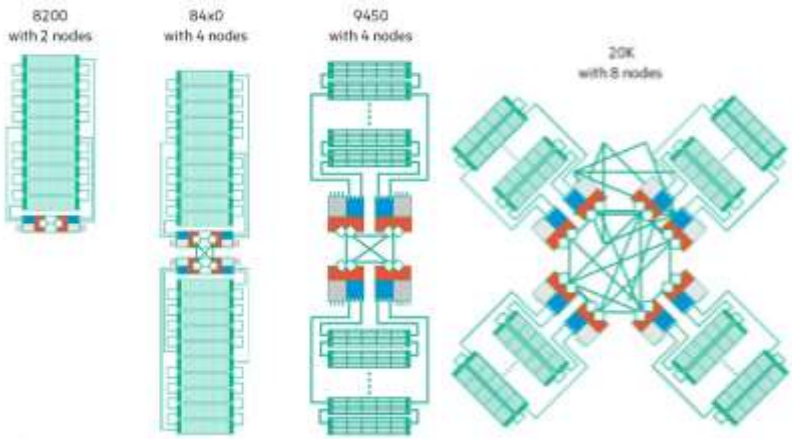
The tri-level abstraction methodology imposed by the HPE 3PAR Operating System relies on a fine-grained virtualization approach that divides each physical disk into granular allocation units referred to as chunklets, each of which can be independently assigned and dynamically reassigned to different logical disks that are used to create virtual volumes.

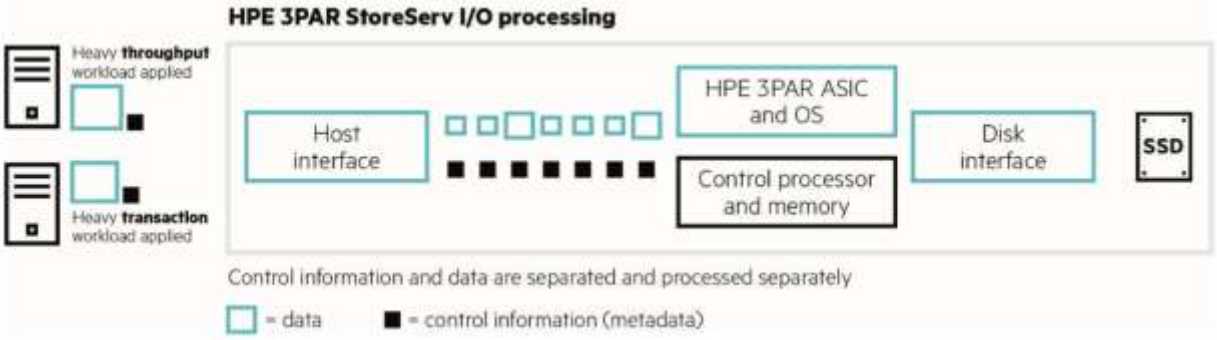
The third layer of abstraction maps LDs to Virtual Volumes (VVs), with all or portions of multiple underlying LDs mapped to the VV. VVs are the virtual capacity representations that are ultimately exported to hosts and applications as virtual LUNs (VLUNs) over Fibre Channel, iSCSI, or FCoE target ports. A single VV can be coherently exported through as few as two ports or as many as ports as desired (no fewer than two, one from each of two different nodes as a minimum). This layer of abstraction uses a table-based association—a mapping table with a granularity of 128 MB per region and an exception table with a granularity of 16 KB per page—as opposed to an algorithmic association. With this approach, a very small portion of a VV associated with a particular LD can be quickly and non-disruptively migrated to a different LD for performance or other policy-based reasons, whereas other architectures require migration of the entire VV. This layer of abstraction also implements many high-level features such as snapshots, caching, pre-fetching, and remote replication.

101. The Accused ’788 Products intermediate I/O subsystem traffic between the one or more application servers and the one or more peripheral systems. As an example, the 3PAR system has a full mesh backplane allowing servers to access volumes over any host-connected port:

HPE 3PAR 20800 offers scalability in its truest sense not only with support for 2304 drives and 9.6 PB raw capacity but also with a port count of up to 160 FC ports for host connectivity and for consuming other advanced data services like replication, host persona, federation or to be used for Storage Class Memory. Each of these ports is connected directly on the I/O bus, so all ports can achieve full bandwidth up to the limit of the I/O bus bandwidths that they share.

In every HPE 3PAR Storage system, each controller node has a dedicated link to each of the other nodes that operates at 4 GiB/s in each direction. In an HPE 3PAR 20800 Storage system, a total of 56 of these links form the array's full-mesh backplane. In addition, each controller node may have one or more paths to hosts—either directly or over a storage area network (SAN). The clustering of controller nodes enables the system to present hosts with a single, highly available, high-performance storage system. This means that servers can access volumes over any host-connected port—even if the physical storage for the data is connected to a different controller node. This is achieved through an extremely low-latency data transfer across the high-speed, full-mesh backplane.





102. The Accused '788 Products control utilization of resources of the I/O subsystem physical interface by the one or more application servers according to a configured allocation of resources for the I/O subsystem physical interface across the one or more application servers. For instance, the HPE 3PAR Priority Optimization software operates on I/O communications between the 3PAR system and a host, and applies priority policies to the communication based on volume or domain, which can include maximum and minimum limits, goals and priority levels:

HPE 3PAR Priority Optimization software for HPE 3PAR StoreServ Storage systems implements and manages a priority policy per virtual volume set (VVset) that serves as a proxy for applications and per virtual domains that serves as a proxy for customers or tenants. Volumes that are not a member of a VVset or a virtual domain can be controlled using the System QoS rule. Priority Optimization enables users to take full control of performance by specifying minimum and maximum limits ("Min Goal" and "Max Limit") for IOPS and bandwidth, as well as a priority for every QoS object along with the ability to define latency goals for the most important applications. If these goals are not met, the system automatically adjusts the service levels of lower-priority applications and workloads in order to make sure that necessary QoS levels for the highest priority applications are always maintained. This paradigm is valid on HPE 3PAR StoreServ systems because all

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.

Priority Optimization sets the Min Goal and Max Limit values for IOPS and bandwidth in QoS rules in absolute numbers, not in percentages. The IOPS number is stated as an integer between zero and 2,147,483,647 (2^{31} - 1), although a more realistic upper limit is the number of IOPS the HPE 3PAR StoreServ is rated for. The bandwidth value is an integer between zero and 9,007,199,254,740,991 (slightly less than 2^{53}) expressed in KB/s, although a more realistic upper limit is the throughput in KB/s the HPE 3PAR StoreServ can handle over its I/O ports.

The latency goal of a QoS rule is the maximum value for the service time the system will try to provide for the applications governed by that rule. For the latency goal to work, rules with a Min Goal must exist so the system knows the minimum to which it can throttle other workloads to achieve the latency goal. Hierarchy in workloads is created by assigning them a priority level of high, normal, and low. As the system gets busier, it starts throttling lower-priority workloads first to meet latency goals of higher-priority workloads. In HPE 3PAR OS 3.2.2 the minimum latency goal was reduced from 1 to 0.5 milliseconds to accommodate the reduced service time seen on HPE 3PAR StoreServ all flash arrays. The latency goal ranges from 0.50 to 10,000 milliseconds.

103. Additionally, HPE has been, and currently is an active inducer of infringement of the ’788 patent under 35 U.S.C. § 271(b) and contributory infringement of the ’788 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

104. HPE has actively induced, and continues to actively induce, infringement of the ’788 patent by intending that others use, offer for sale, sell, and/or import products and/or services covered by the ’788 patent, including but not limited to the Accused ’788 Products and any other HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States products and/or services that directly infringe the ’788 patent.

105. HPE has contributed to, and continues to contribute to, the infringement of the ’788 patent by others by offering to sell, selling, or otherwise commercially offering products and/or services that, when installed and configured result in a system as intended by HPE, that directly infringes the ’788 patent.

106. HPE knew of the ’788 patent, or should have known of the ’788 patent, but was willfully blind to its existence. HPE has had actual knowledge of the ’788 patent since at least as early as the filing of this Complaint. Additionally, HPE was aware of the ’788 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the ’788 patent.

107. HPE has committed, and continues to commit, affirmative acts that cause infringement of the ’788 patent with knowledge of the ’788 patent and knowledge or willful blindness that the induced acts constitute infringement of the ’788 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customer way as desired and intended by HPE, infringe the ’788 patent and/or by directly or indirectly providing instruction on how to use its products and/or services in a manner or configuration that infringes the ’788 patent, including those found at the following:

- https://www.hpe.com/us/en/storage/3par.html
- https://h20195.www2.hpe.com/v2/GetPDF.aspx/4AA4-7264ENW.pdf
- https://h20195.www2.hpe.com/v2/getdocument.aspx?docname=a00073435enw
- https://h20195.www2.hpe.com/v2/getpdf.aspx/4aa3-3516enw.pdf
- https://support.hpe.com/hpesc/public/docDisplay?docLocale=en_US&docId=emr_na-a00067530en_us
- https://www.hpe.com/us/en/storage/hpe-primera.html

  https://www.hpe.com/us/en/greenlake/enterprise-ready-vms.html
- https://www.hpe.com/us/en/greenlake/mission-critical-storage.html
- https://psnow.ext.hpe.com/doc/a50000189enw

108. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that, when used, cause the direct infringement of the '788 patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such as software, that are especially made or especially adapted for use in infringement of the '788 patent and are not staple articles or commodities of commerce suitable for substantial non-infringing use.

109. As a result of HPE's acts of infringement, IV has suffered and will continue to suffer damages in an amount to be paid at trial.

COUNT IV
(HPE's Infringement of U.S. Patent No. RE 44,818)

110. Paragraphs 1-109 are incorporated by reference as if fully set forth herein.

111. The inventions claimed by the RE '818 patent, taken alone or in combination, were not well-understood, routine or conventional to one of ordinary skill in the art at the time of the invention of the RE '818 patent, around 2007. Rather, the patent teaches and claims an improved way to facilitate and manage virtualized input/output ("I/O") subsystems (*e.g.*, virtual I/O servers) that represented a novel and non-obvious approach not present in the state of the art at that time. The inventions improved upon then-existing virtual I/O servers, which are continuously engaged in network communications, by for example, enabling more granular QoS (Quality of Service) controls to be applied to those communications. The inventions further improved prior art virtual I/O systems by using a hierarchical token bucket allocator to determine how resources are allocated to I/O communications received by virtual I/O servers. Specifically, the inventions require maintaining a connection over a network fabric between a virtual interface layer of an application server and a virtual I/O server to receive I/O communications, presenting a virtual node identifier to the virtual I/O server at a physical interface, whereupon receiving the I/O communications at

the virtual I/O server, enforcing I/O bandwidth limitations via a hierarchical token bucket resource allocation that attaches token allocations to an I/O communication, such that the I/O packets are only received at the virtual I/O server and forwarded to the virtualized I/O systems connected to the virtual l/O server if there is a sufficient amount of remaining tokens attaching to that I/O communication.

112. This is to be contrasted with then-existing systems that, for example, did not enforce more granular QoS (Quality of Service) requirements on the interfaces of virtual I/O servers as they engaged in I/O operations. This resulted in an inefficient allocation of network resources to such virtual I/O server communications, which in turn lead to suboptimal I/O subsystem performance. Virtual I/O servers, for example, were not previously able to apply tailored virtualized I/O allocations to servers, as is now required to handle the wide variety of I/O communications typical of today's highly virtualized environments.

113. The inventions represented a technical solution to an unsolved technological problem. The specification of the RE '818 patent describes, in technical detail, each of the limitations in the claims, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claims differ markedly from what had been conventional or generic in the industry at the time of the invention of the RE '818 patent. More specifically, the claims of the RE '818 patent require maintaining a connection over a network fabric to a virtual network interface layer of an application server, presenting, at a physical interface, a virtual node identifier to the input output subsystem, enforcing a hierarchical token bucket resource allocation of bandwidth across the physical interface, receiving, over the connection, an I/O communication to a target on the I/O subsystem, classifying the received I/O

communication relative to the hierarchical token bucket resource allocation to determine the amount of tokens available, comparing the size of the received I/O communication to the current amount of tokens available, forwarding the received I/O communication across the physical interface to the I/O subsystem, if the current amount of tokens available are sufficient, and buffering the received I/O communication, if the current amount of tokens available are insufficient.

114. The systems and methods covered by the asserted claims therefore differ markedly from the conventional and generic systems in use at the time of this invention, which *inter alia* lacked the ability to use a hierarchical token bucket allocator to determine how resources are allocated to I/O communications received by virtual I/O servers. Such functionality, for instance, in turn enables resource allocation decisions that meet QoS (Quality of Service) requirements to be made at multiple points/layers in the virtual I/O system (e.g., I/O allocation decisions can be made with respect to a type of traffic, or with respect to traffic moving to/from a server associated with a specific user).

115. As described above, the RE ’818 patent is drawn to solving a specific, technical problem arising in the context of facilitating and managing I/O communications between a virtual I/O server and virtual interface of an application server, over a physical interface in a virtualized environment. Consistent with the problem addressed being rooted in I/O communication techniques in a virtualized environment, the RE ’818 patent’s solutions naturally are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

116. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 32 of the RE ’818 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered

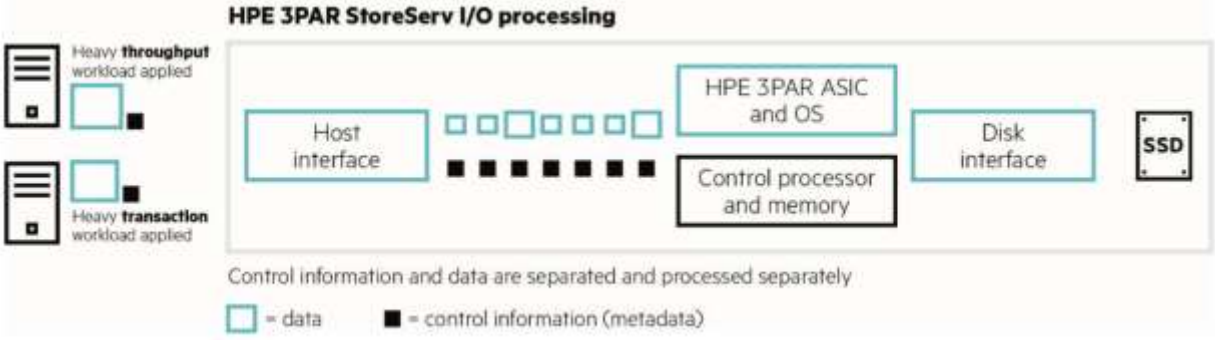
by the RE '818 patent. HPE's products and/or services that infringe the RE '818 patent include, but are not limited to, the HPE 3PAR StoreServ family of flash-optimized data storage systems, the HPE Primera Storage, HPE GreenLake Enterprise-Ready VM Service when implemented with Primera Storage platform, HPE GreenLake Mission Critical Storage Service when implemented with Primera Storage platform, and any other of HPE's products and/or services, either alone or in combination, that operate in substantially the same manner (together the "Accused '818 Products").

Claim 32 of the RE '818 patent is reproduced below:

> *32. A method of facilitating management of input/output subsystems in a virtual input/output server, the method comprising:*
> *maintaining a connection, over a network fabric, to a virtual interface layer of an application server, to receive input/output communications to an input/output subsystem;*
> *presenting, at a physical interface, a virtual node identifier to the input/output subsystem;*
> *enforcing a hierarchical token bucket resource allocation of bandwidth across the physical interface;*
> *receiving, over the connection, an input/output communication to a target on the input/output subsystem, thereby resulting in received input/output communication;*
> *classifying the received input/output communication relative to the hierarchical token bucket resource allocation to determine a current amount of tokens available;*
> *comparing a size of the received input/output communication to the current amount of tokens available;*
> *forwarding the received input/output communication across the physical interface to the input/output subsystem, if the current amount of tokens available are sufficient; and*
> *buffering the received input/output communication, if the current amount of tokens available are insufficient.*

117. As one non-limiting example, the Accused '818 Products provide for facilitating management of input/output subsystems in a virtual input/output server by deploying the HPE 3PAR Storage highly virtualized storage operating system, a tri-level mapping system with three

layers of abstraction and priority optimization settings applied to I/O communications as seen below:



Priority Optimization

The HPE 3PAR architecture is capable of extremely high performance—up to 10 million IOPS in a four-system federation.[2] However, there are few applications that can take advantage of even hundreds of thousands of IOPS on their own. The performance and capacity scalability of the HPE 3PAR platform combined with the plethora of mixed workload and multi-tenancy features makes it the ideal destination platform to use when consolidating multiple enterprise workloads to a single array. While many applications can share system resources without incident, many enterprise applications there are some applications and workloads that are capable of runaway performance demands that can starve other applications on the same platform that are competing for performance resources.

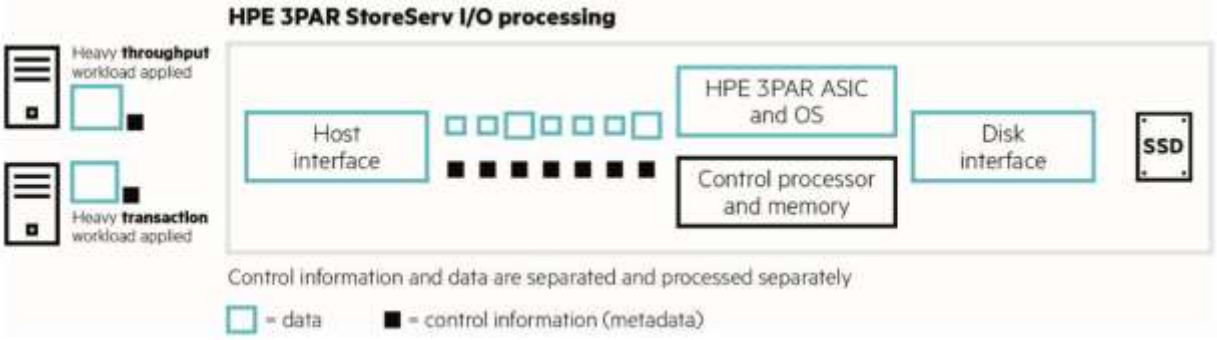
HPE 3PAR Priority Optimization assures quality of service (QoS) by enabling customers to guarantee service levels in not only multi-tenant environments but any environment that hosts multiple services. It enables customers to define maximum and/or minimum performance thresholds for front-end IOPS and/or bandwidth with application-level granularity. Customers can also set latency targets as low as 500 microseconds for volume sets and help ensure that their most critical applications get the right resources to meet their service level targets (Figure 6).

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

118. The Accused ’818 Products also provide for maintaining a connection, over a network fabric, to a virtual interface layer of an application server, to receive input/output communications to an input/output subsystem. For instance, the HPE 3PAR system interfaces with virtualized application servers (which run on “virtualization platforms”, which in turn are deployed on “hosts”) via a network, such as a LAN, over which I/O communications like read/write requests are passed to and from the virtualized storage that the Accused Products provide:



Virtualization software

Virtualization platforms such as VMware®, Microsoft® Hyper-V®, and Citrix® XenServer® execute business-critical and latency-sensitive applications in parallel with other less-critical applications. It is important to prioritize I/O requests from important applications over the requests from less important applications. The virtualization software creates and manages virtual data stores that are container files comprising one or more virtual drives used by virtual machines (VMs). Each container file is built on one or more storage LUNs that reside on the storage array and are usually accessed over the SAN. Applying a Priority Optimization QoS rule to the LUNs that make up a logical storage container will control the IOPS or bandwidth for all VMs whose virtual drives are contained in that LUN. Be sure that there is enough bandwidth and IOPS in the QoS rule so that the applications running on the VMs can deliver acceptable I/O response times.

A QoS rule on a VVset making up a VMware vSphere® datastore puts a limit on the I/O requests for all VMs using that datastore. This level of control is not granular enough if you want to grant some VMs more I/O resources than others in case of contention. Recent versions of vSphere offer three native types of granular I/O resource control. Table 3 lists the vSphere characteristics together with those of Priority Optimization QoS.

Virtual volume sets and virtual domains

QoS rules operate on VVsets, virtual domains, and volumes that belong to neither of them. A VVset is an autonomic group object that is a collection of virtual volumes. VVsets help to simplify the administration of virtual volumes and reduce human error. An operation like exporting a VVset to a host will export all member volumes of the VVset. Adding a volume to an existing VVset will export that volume automatically to the host or host set the VVset is exported to. VVsets have several use cases beyond reducing the administration of their volume members, such as enabling simultaneous point-in-time snapshots of a group of volumes with a single command. Up to 8,192 volumes can be part of a VVset and up to 8,192 VVsets can be created. There is no requirement that volumes in a VVset are exported to a host for a QoS rule to be applied to the VVset.

Virtual volumes overview

Virtual volumes are created from pools of storage called common provisioning groups and can be exported to hosts. Virtual volumes are the only data layer in HPE 3PAR storage virtualization that is visible to hosts.

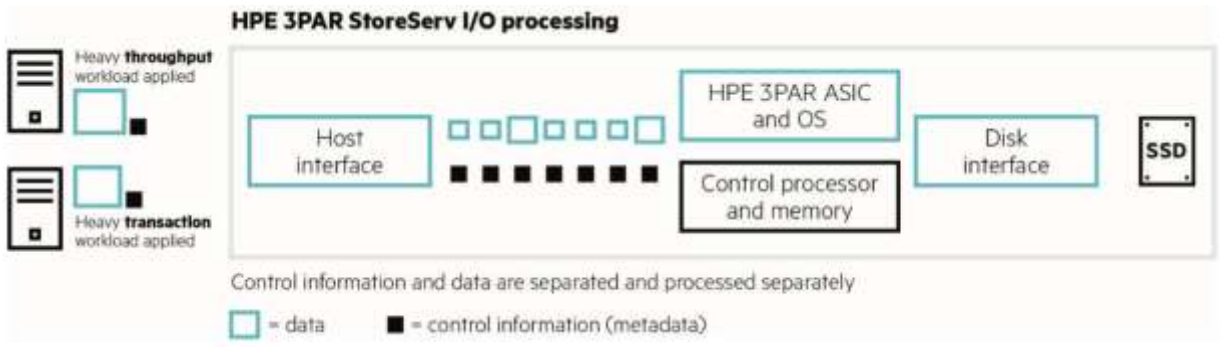
Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

119. Furthermore, the Accused ’818 Products facilitate presenting, at a physical interface, a virtual node identifier to the input/output subsystem. Following the above example,

the HPE 3PAR system includes a host interface and can receive I/O communications via iSCSI or Fibre Channels, that support the iSCSI message format in which "initiators" and "targets" are identified via a unique name, such as an iSCSI qualified name (IQN), an extended-unique identifier (EUI) or worldwide node name (WWNN) for Fibre Channels. Alternatively, or in addition, virtual volumes necessarily include a unique identifier:



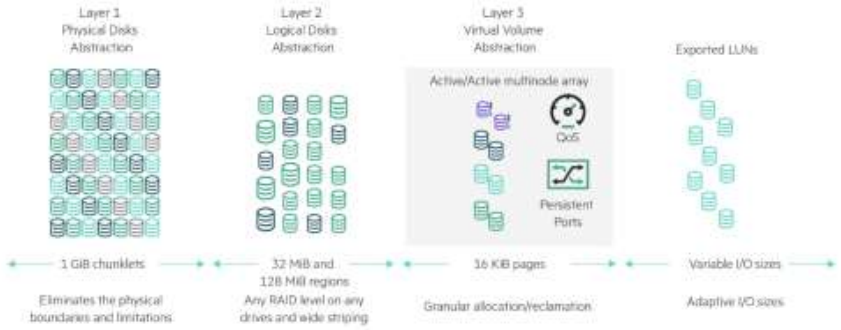
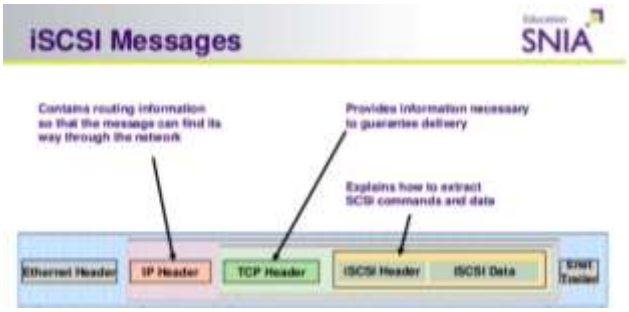


FIGURE 5. Virtualization with a tri-level mapping methodology that provides three layers of abstraction

Each HPE 3PAR 8000-series controller node has two built-in 4 port 16 Gbps Fibre Channel ports and one PCIe expansion slot. This slot can hold one of the following adapters:

- Dual-port 32 Gbps Fibre Channel adapter
- Quad-port 16 Gbps Fibre Channel adapter
- Dual-port 10 Gbps iSCSI or Fibre Channel over Ethernet (FCoE) converged network adapter
- Quad-port 16 Gb Fibre Channel/10 Gb NIC Combo Adapter
- Quad-port 10 Gb iSCSI/10 Gb NIC Combo Adapter
- Dual-port 10 Gbps Ethernet adapter for file and object access services using HPE 3PAR File Persona
- Quad-port 1 Gbps Ethernet adapter for file and object access services using HPE 3PAR File Persona



iSCSI initiators and targets

In an iSCSI configuration, the iSCSI host or server sends requests to a node. The host contains one or more initiators that attach to an IP network to initiate requests to and receive responses from an iSCSI target. Each initiator and target are given a unique iSCSI name such as an iSCSI qualified name (IQN) or an extended-unique identifier (EUI). An IQN is a 223-byte ASCII name. An EUI is a 64-bit identifier. An iSCSI name represents a worldwide unique naming scheme. This scheme identifies each initiator or target in the same way that worldwide node names (WWNNs) are used to identify devices in a Fibre Channel fabric.

Virtual volumes overview

Virtual volumes are created from pools of storage called common provisioning groups and can be exported to hosts. Virtual volumes are the only data layer in HPE 3PAR storage virtualization that is visible to hosts.

120. The Accused ’818 Products also enable enforcing a hierarchical token bucket resource allocation of bandwidth across the physical interface. For instance, the HPE 3PAR system enforces Priority Optimization on I/O communications to the virtual storage volumes it provides, which allows for applying one or more priority policies in a hierarchical manner to virtual volumes/applications and virtual domains/tenants, and additionally allows for the application of secondary QoS rules to individual virtual volumes or groups of them:

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

HPE 3PAR Priority Optimization software for HPE 3PAR StoreServ Storage systems implements and manages a priority policy per virtual volume set (VVset) that serves as a proxy for applications and per virtual domains that serves as a proxy for customers or tenants. Volumes

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.

Priority Optimization supports nesting QoS rules. This allows for the creation of secondary QoS rules on individual VVsets or groups of them in the domain of the tenant. These nested rules limit I/O for smaller objects within a tenant's domain. Nested QoS rules can be overprovisioned without the risk of adversely affecting the workloads of other tenants because the top-level QoS rule enforces a Max Limit on the I/O capacity of all objects within the tenant's domain.

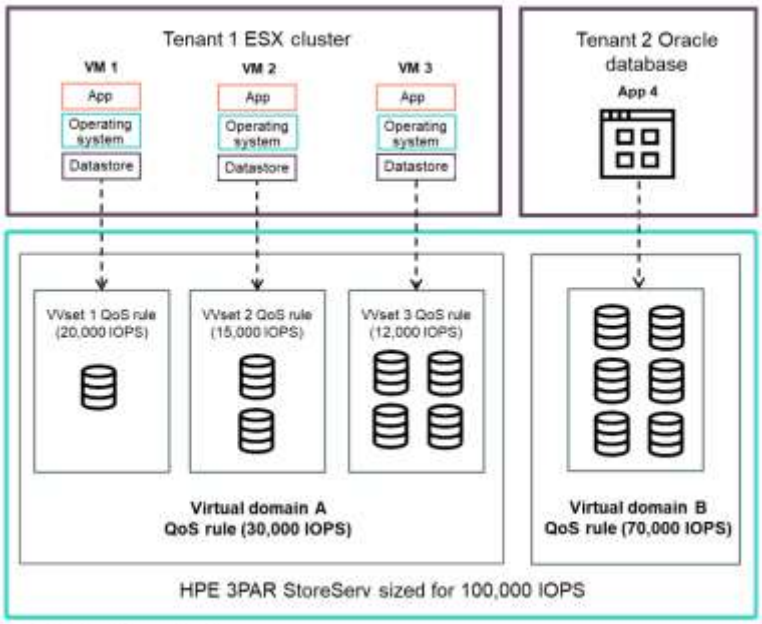


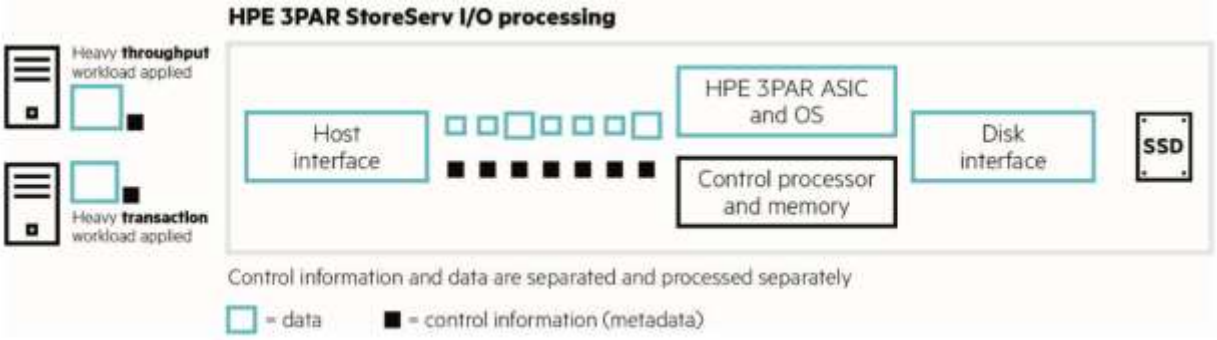
Figure 21. Top-level and nested QoS rules

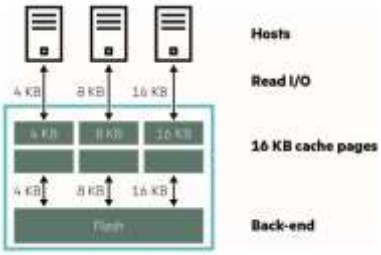
121. The Accused '818 Products further enable receiving an input/output communication to a target on the input/output system. The HPE 3PAR system provides virtual storage for virtual/application servers such that the I/O communications are received, for example, at the 3PAR system's controllers:

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request





122. The Accused '818 Products also enable classifying the received I/O communication relative to the hierarchical token bucket resource allocation to determine a current amount of tokens available. As an example, the 3PAR system's Priority Optimization functionality operates on I/O communications, classifying each based on the set policy which is applied at the virtual volume level and the virtual domain level. The policies include, for instance, a maximum limit, minimum limit and other customizable goals/priority levels that represent portions of the overall IOPS bandwidth:

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

If the upper limit for IOPS or bandwidth for a VVset has been reached, Priority Optimization delays I/O request responses for the VVs contained in that VVset. These delayed I/O requests are pushed to an outstanding I/O queue for the volumes in the VVset experiencing the limit breach. Every QoS rule maintains its own queue for delayed I/O requests. These queues are constructed in the control cache of the HPE 3PAR StoreServ controller node receiving the I/O request that needs to be delayed. Only the I/O request (SCSI read or write request) descriptions are queued, not the actual data, leading to very fast handling of delay operations and no data cache consumption.

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.

Priority Optimization sets the Min Goal and Max Limit values for IOPS and bandwidth in QoS rules in absolute numbers, not in percentages. The IOPS number is stated as an integer between zero and 2,147,483,647 (2^{31} - 1), although a more realistic upper limit is the number of IOPS the HPE 3PAR StoreServ is rated for. The bandwidth value is an integer between zero and 9,007,199,254,740,991 (slightly less than 2^{63}) expressed in KB/s, although a more realistic upper limit is the throughput in KB/s the HPE 3PAR StoreServ can handle over its I/O ports.

The latency goal of a QoS rule is the maximum value for the service time the system will try to provide for the applications governed by that rule. For the latency goal to work, rules with a Min Goal must exist so the system knows the minimum to which it can throttle other workloads to achieve the latency goal. Hierarchy in workloads is created by assigning them a priority level of high, normal, and low. As the system gets busier, it starts throttling lower-priority workloads first to meet latency goals of higher-priority workloads. In HPE 3PAR OS 3.2.2 the minimum latency goal was reduced from 1 to 0.5 milliseconds to accommodate the reduced service time seen on HPE 3PAR StoreServ all flash arrays. The latency goal ranges from 0.50 to 10,000 milliseconds.

123. The Accused ’818 Products enable comparing a size of the received I/O communication to the current amount of tokens available. For instance, if the maximum allocated bandwidth for a volume or domain has been reached then the I/O communication will be queued, however, if there is sufficient bandwidth remaining unallocated then the I/O communication can be forwarded accordingly:

Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

If the upper limit for IOPS or bandwidth for a VVset has been reached, Priority Optimization delays I/O request responses for the VVs contained in that VVset. These delayed I/O requests are pushed to an outstanding I/O queue for the volumes in the VVset experiencing the limit breach. Every QoS rule maintains its own queue for delayed I/O requests. These queues are constructed in the control cache of the HPE 3PAR StoreServ controller node receiving the I/O request that needs to be delayed. Only the I/O request (SCSI read or write request) descriptions are queued, not the actual data, leading to very fast handling of delay operations and no data cache consumption.

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.

HPE 3PAR Priority Optimization software for HPE 3PAR StoreServ Storage systems implements and manages a priority policy per virtual volume set (VVset) that serves as a proxy for applications and per virtual domains that serves as a proxy for customers or tenants. Volumes that are not a member of a VVset or a virtual domain can be controlled using the System QoS rule. Priority Optimization enables users to take full control of performance by specifying minimum and maximum limits ("Min Goal" and "Max Limit") for IOPS and bandwidth, as well as a priority for every QoS object along with the ability to define latency goals for the most important applications. If these goals are not met, the system automatically adjusts the service levels of lower-priority applications and workloads in order to make sure that necessary QoS levels for the highest priority applications are always maintained. This paradigm is valid on HPE 3PAR StoreServ systems because all

124. The Accused '818 Products enable forwarding the received I/O communication across the physical interface to the I/O subsystem if the current amount of available tokens is sufficient. Continuing with the above example, the 3PAR system will forward the I/O communication downstream from the host interface towards storage when the system determines that the appropriate policy limit for allocated bandwidth for that type of traffic is sufficient, otherwise, the communication will be queued:

Mode of operation

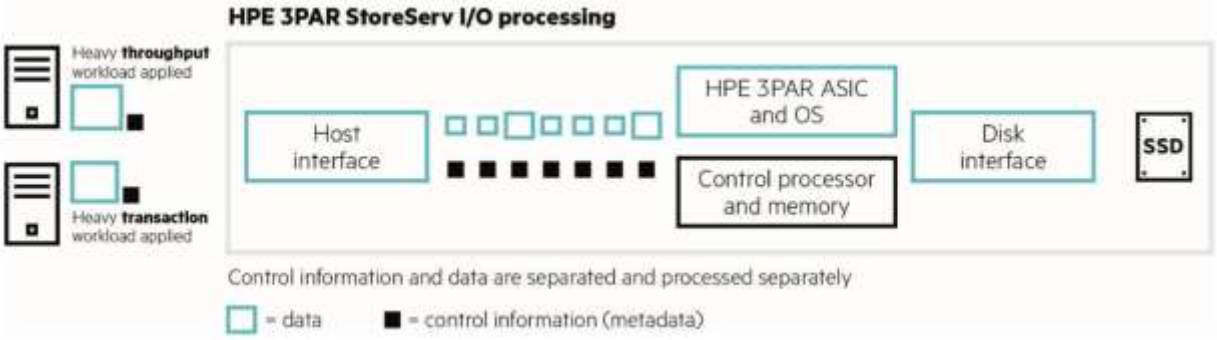
Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

If the upper limit for IOPS or bandwidth for a VVset has been reached, Priority Optimization delays I/O request responses for the VVs contained in that VVset. These delayed I/O requests are pushed to an outstanding I/O queue for the volumes in the VVset experiencing the limit breach. Every QoS rule maintains its own queue for delayed I/O requests. These queues are constructed in the control cache of the HPE 3PAR StoreServ controller node receiving the I/O request that needs to be delayed. Only the I/O request (SCSI read or write request) descriptions are queued, not the actual data, leading to very fast handling of delay operations and no data cache consumption.

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.

HPE 3PAR Priority Optimization software for HPE 3PAR StoreServ Storage systems implements and manages a priority policy per virtual volume set (VVset) that serves as a proxy for applications and per virtual domains that serves as a proxy for customers or tenants. Volumes that are not a member of a VVset or a virtual domain can be controlled using the System QoS rule. Priority Optimization enables users to take full control of performance by specifying minimum and maximum limits ("Min Goal" and "Max Limit") for IOPS and bandwidth, as well as a priority for every QoS object along with the ability to define latency goals for the most important applications. If these goals are not met, the system automatically adjusts the service levels of lower-priority applications and workloads in order to make sure that necessary QoS levels for the highest priority applications are always maintained. This paradigm is valid on HPE 3PAR StoreServ systems because all



125. The Accused ’818 Products enable the buffering of the I/O communication if the current amount of available tokens is insufficient. Continuing with the above example, the communication will be queued if the policy limit for allocated bandwidth for that traffic type is exceeded:

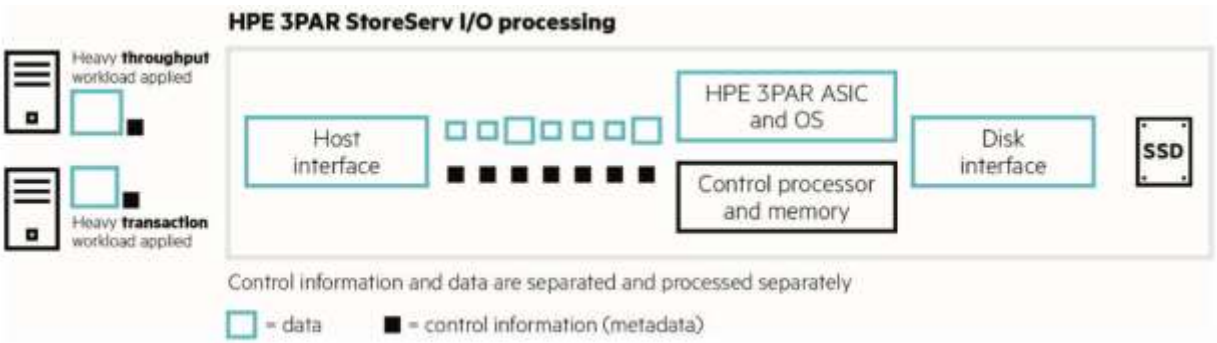
Mode of operation

Priority Optimization operates on I/O communication between an HPE 3PAR StoreServ and a host. The QoS rules are defined for front-end IOPS and bandwidth and are applied via VVsets and virtual domains. When an I/O request reaches the HPE 3PAR StoreServ controllers, Priority Optimization will take one of the following three actions:

- Pass the I/O request to the VV layer of the HPE 3PAR OS
- Delay the I/O request by placing it in a private QoS queue on the HPE 3PAR StoreServ and process the queue periodically
- Return an SCSI Queue Full (QFULL) message to the server that sent the request

If the upper limit for IOPS or bandwidth for a VVset has been reached, Priority Optimization delays I/O request responses for the VVs contained in that VVset. These delayed I/O requests are pushed to an outstanding I/O queue for the volumes in the VVset experiencing the limit breach. Every QoS rule maintains its own queue for delayed I/O requests. These queues are constructed in the control cache of the HPE 3PAR StoreServ controller node receiving the I/O request that needs to be delayed. Only the I/O request (SCSI read or write request) descriptions are queued, not the actual data, leading to very fast handling of delay operations and no data cache consumption.

Starting from HPE 3PAR OS 3.1.2 MU2, Priority Optimization can cap performance of VVsets with an upper IOPS, bandwidth limit, or both. This cap is called the **Max Limit** and is the maximum amount of IOPS, bandwidth, or both that a given QoS object is allowed to achieve. HPE 3PAR System Reporter can be used to quantify the volume performance upon which acceptable QoS rules can be defined. HPE 3PAR OS 3.1.3 introduced the concepts of a Min Goal, a priority level (high, normal, and low), and a latency goal per QoS object along with the ability to define QoS rules against virtual domains. The Min Goal is the minimum amount of IOPS or bandwidth that the QoS subsystem will throttle a given QoS object in order to meet the latency goal of a higher-priority workload. If system resources are available, VVsets or virtual domains might consume more IOPS or bandwidth than the Min Goal but will be throttled if they reach their Max Limit. The performance might also be less than the Min Goal; this can happen if the application is requesting fewer IOPS or if the sum of all Min Goals defined is more than the I/O capability of the system or a given tier of storage.



126. Additionally, HPE has been, and currently is an active inducer of infringement of the RE ’818 patent under 35 U.S.C. § 271(b) and contributory infringement of the RE ’818 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

127. HPE has actively induced, and continues to actively induce, infringement of the RE ’818 patent by intending that others use, offer for sale, sell and/or import products and/or services covered by the RE ’818 patent, including but not limited to the Accused ’818 Products and any other HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States products and/or services that directly infringe the RE ’818 patent.

128. HPE has contributed to, and continues to contribute to, the infringement of the RE ’818 patent by others by offering to sell, selling, or otherwise commercially offering products and/or services that, when installed and configured result in a system as intended by HPE, that directly infringes the RE ’818 patent.

129. HPE knew of the RE ’818 patent, or should have known of the RE ’818 patent, but was willfully blind to its existence. HPE has had actual knowledge of the RE ’818 patent since at least as early as the filing of this Complaint. On Information and belief, HPE had actual knowledge of the RE ’818 patent in and around July 2019. On information and belief, HPE was aware of IV’s allegations against VMware Inc. relating to infringement of the RE ’818 patent by VMware’s deployment of input/output subsystems in a virtual input/output server. Additionally, HPE was aware of the RE ’818 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice)

that its continued actions would infringe and actively induce and contribute to the infringement of the RE ’818 patent.

130. HPE has committed, and continues to commit, affirmative acts that cause infringement of the RE ’818 patent with knowledge of the RE ’818 patent and knowledge or willful blindness that the induced acts constitute infringement of the RE ’818 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customer way as desired and intended by HPE, infringe the RE ’818 patent and/or by directly or indirectly providing instruction on how to use its products and/or services in a manner or configuration that infringes the RE ’818 patent, including those found at the following:

- https://www.hpe.com/us/en/storage/3par.html
- https://h20195.www2.hpe.com/v2/GetPDF.aspx/4AA4-7264ENW.pdf
- https://h20195.www2.hpe.com/v2/getdocument.aspx?docname=a00073435enw
- https://h20195.www2.hpe.com/v2/getpdf.aspx/4aa3-3516enw.pdf
- https://support.hpe.com/hpesc/public/docDisplay?docLocale=en_US&docId=emr_na-a00067530en_us
- https://www.hpe.com/us/en/storage/hpe-primera.html
- https://www.hpe.com/us/en/greenlake/enterprise-ready-vms.html
- https://www.hpe.com/us/en/greenlake/mission-critical-storage.html
- https://psnow.ext.hpe.com/doc/a50000189enw

131. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that, when used, cause the direct infringement of the RE ’818 patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such

as software, that are especially made or especially adapted for use in infringement of the RE '818 patent and are not a staple article or commodity of commerce suitable for substantial non-infringing use.

132. As a result of HPE's acts of infringement, IV has suffered and will continue to suffer damages in an amount to be paid at trial.

COUNT V
(HPE's Infringement of U.S. Patent No. 6,816,464)

133. Paragraphs 1-132 are incorporated by reference as if fully set forth herein.

134. The inventions claimed by the '464 patent, taken alone or in combination, were not well-understood, routine or conventional to one of ordinary skill in the art at the time of the invention. Rather, the '464 patent claims and teaches, *inter alia*, improved ways to assess candidate network routes when establishing a communications link across a network such as a wide area network (WAN), which were not present in the state of the art at the time of the invention. For example, the inventions improved upon existing route assessment techniques by providing for automatic route monitoring, scoring, and evaluation on a per communication link basis and in real time. The inventions can further implement user-customizable routing preferences in the course of providing said route scoring and evaluation, allowing for fine-grained user control over the routing process.

135. Compared to the prior art, the claimed approach implements improved routing intelligence by providing route monitoring, scoring, and evaluation, on a per communications link basis, of a plurality of candidate routes that automatically accounts for jitter, delay, or dropped traffic by taking real-time measurements, while also considering user-customizable routing preferences. This in turn allows for improved quality of service and dynamic traffic shaping capabilities not possible in prior art solutions.

136. The inventions represented a technical solution to an unsolved technological problem. The specification of the '464 patent describes, in technical detail, each of the limitations in the claims, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claims differ markedly from what had been conventional or generic in the industry at the time of the inventions of the '464 patent. More specifically, the claims of the '464 patent recite methods and systems for assessing network routes for use in establishing a communications link including identifying candidate routes and associated terminal gateway(s), transmitting quality measurement packets to determine route quality metrics on each candidate route, and receiving quality measurement packets to determine route quality statistics from each candidate route. The claimed systems and methods further recite determining route statistics based on routing information in the quality measurement packets, configuring a route ordering schedule based on user input, and scoring the candidate routes based on the routing statistics and the route ordering schedule to configure a scoring table that includes a quality score and packet loss, jitter, and/or delay.

137. The systems covered by the asserted claims, therefore, differs markedly from the conventional and generic systems in use at the time of this invention, which, *inter alia*, lacked the claimed combination of identifying a plurality of candidate routes, transmitting and receiving quality measurement packets, determining route statistics, configuring a route ordering schedule based on user input, and scoring the candidate routes to configure a scoring table that includes a quality score and packet loss, jitter, and/or delay. Embodiments of the present inventions further teach operation across packet-switched networks (such as the Internet); inclusion of databases that store and allow for consideration of historical routing information; graphical user interfaces

("GUIs") for accepting user routing preferences; and advanced settings for configuring route measurement properties, timings, and statistical analysis.

138. As described above, the '464 patent is drawn to solving a specific, technical problem arising in the context of loss, latency and jitter sensitive route selection in packet-switched networks. Consistent with the problem addressed being rooted in such packet-switched network environments, the solutions provided by the '464 patent consequently are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

139. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 1of the '464 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the '464 patent. HPE's products and/or services that infringe the '464 patent include, but are not limited to, HPE's Silver Peak Unity EdgeConnect SD-WAN Edge Platform, including Intelligent Internet Breakout, a/k/a Aruba EdgeConnect Platform, ("Unity EdgeConnect"), and any other HPE products and/or services, either alone or in combination, that operate in substantially the same manner (together the "Accused '464 Products").

Claim 1 of the '464 patent is reproduced below:

> *1. A method for assessing network routes for use in establishing a communications link within a communications network, comprising the steps of:*
> *(1) identifying a plurality of candidate routes that can be used to establish said communication link, wherein a terminating gateway associated with each of said plurality of candidate routes is identified;*
> *(2) transmitting quality measurement packets for each of said candidate routes, wherein said quality measurement packets can be used to determine at least one route quality metric;*
> *(3) receiving returned quality measurement packets for each of said candidate routes, wherein said returned quality measurement packets can be used to determine route statistics;*
> *(4) determining route statistics, wherein said route statistics are based on routing information contained within said quality measurement packets;*

(5) configuring a route ordering schedule based on user set levels of route characteristics; and

(6) scoring each of said candidate routes based on route statistics and said route ordering schedule, wherein a scoring table is configured that includes a quality score and one or more of packet loss, average delay, and average jitter.

140. As one non-limiting example, the Accused '464 Products practice a method for assessing network routes for use in establishing a communications link within a communications network. For example, Unity EdgeConnect continuously monitors the performance of all links in a communications network, continuously measures packet loss, jitter, latency, and mean opinion score (MOS) in real-time, and uses statistical learning to dynamically determine which link is performing best and selects that link for sending data traffic. Also, for example, the Unity Orchestrator enables configuration of automated policies for finding the best path for traffic over a network's SD-WAN fabric:

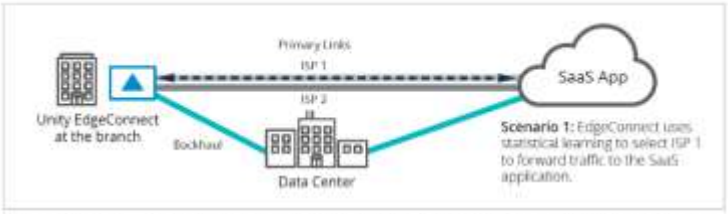


Figure 11: To optimize utilization of the provisioned WAN internet links (ISP 1 and ISP 2), EdgeConnect monitors the performance of the two links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. In this example, based on statistical learning, EdgeConnect dynamically selects ISP 1 to send traffic to the SaaS application since it is performing better than the ISP 2 service.

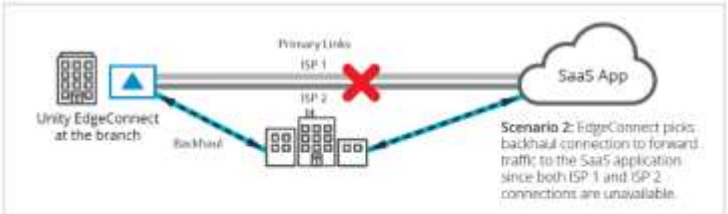


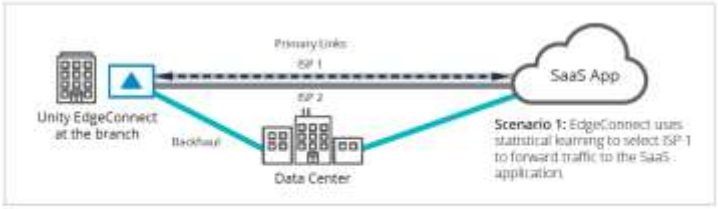
Figure 12: If both ISP 1 and ISP 2 connections become unavailable, EdgeConnect automatically moves application traffic to the transport service configured as a backup that backhauls traffic through the data center.

Intelligent Internet Breakout

Often customers provision two or more WAN links from remote branch sites to increase network and application availability and performance. These links are used for breaking out traffic locally at each branch. Using the internet as an underlay transport is less expensive than private leased line connections such as MPLS since it offers much higher bandwidth at a given price point. To optimize utilization of the provisioned WAN internet links and to optimize SaaS application performance, EdgeConnect monitors the performance of all links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. EdgeConnect uses statistical learning based on jitter, latency, loss and MOS on all provisioned internet links to dynamically determine which link is performing the best before sending traffic. This optimizes internet break out traffic to deliver the highest SaaS and cloud application performance (see Figure 11). Configuring these policies is fully automated within Silver Peak Unity Orchestrator™ and doesn't require any manual configuration. The Orchestrator also enables configuration of an automated policy for finding the best path for that traffic over the SD-WAN fabric, across MPLS or another WAN service, in the rare case that both underlying internet links are underperforming or are unavailable (see Figure 12).

141. The Accused '464 Products practice the step of identifying a plurality of candidate routes that can be used to establish said communication link, wherein a terminating gateway

associated with each of said plurality of candidate routes is identified. For example, the Accused '464 Products identify two or more WAN routes from remote branch sites which are used for communication. The Accused '464 Products also enable the creation of tunnels in order to optimize traffic, identify and label WAN internet routes or tunnels, and identify gateways for each candidate route, such as Unity Orchestrator's use of the TGNM WAN API to target the branches in the network and associate them to a Transit Gateway, configuring the tunnel endpoints for each branch:



Intelligent Internet Breakout

Often customers provision two or more WAN links from remote branch sites to increase network and

Figure 11: To optimize utilization of the provisioned WAN Internet links (ISP 1 and ISP 2), EdgeConnect monitors the performance of the two links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. In this example, based on statistical learning, EdgeConnect dynamically selects ISP 1 to send traffic to the SaaS application since it is performing better than the ISP 2 service.

WAN-side Configuration

Firewall Zone: Zone-based firewalls are created on the Orchestrator. A zone is applied to an **interface**. By default, traffic is allowed between interfaces labeled with the same zone. Any traffic between interfaces with different zones is dropped. Users can create exception rules (Security Policies) to allow traffic between interfaces with different zones. The firewall zones you have already configured will be in the list under **FW Zone**. Select the FW zone you want to apply to the WAN you are deploying.

Firewall Mode: Four options are available at each WAN interface:

- **Allow All** permits unrestricted communication.
- **Stateful** *only* allows communication from the LAN-side to the WAN-side.

  Use this if the interface is behind the WAN edge router.
- **Stateful with SNAT** applies Source NAT to outgoing traffic.

  Use this if the interface is directly connected to the Internet.
- **Harden**
  - For traffic inbound from the WAN, the appliance accepts *only* IPSec tunnel packets that terminate on a Silver Peak appliance.
  - For traffic outbound to the WAN, the appliance *only* allows IPSec tunnel packets and management traffic that terminate on a Silver Peak appliance.

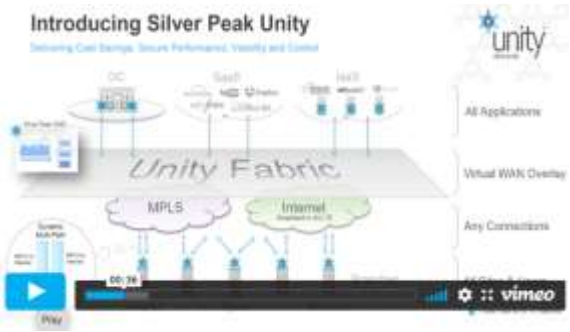
WARNING: Activating fail-to-wire will DISABLE ALL firewall rules.

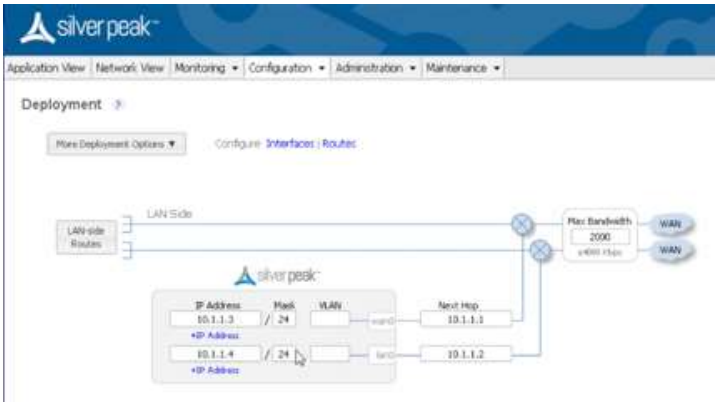
NAT Settings: When using NAT, use in-line Router mode to ensure that addressing works properly. That means you configure paired single or dual WAN and LAN interfaces on the appliance. Select one of the following options:

- If the appliance is behind a NAT-ed interface, select **NAT**.
- If the appliance is not behind a NAT-ed interface, select **Not behind NAT**.
- **Enter an IP address** to assign a destination IP for tunnels being built from the network to this WAN interface.

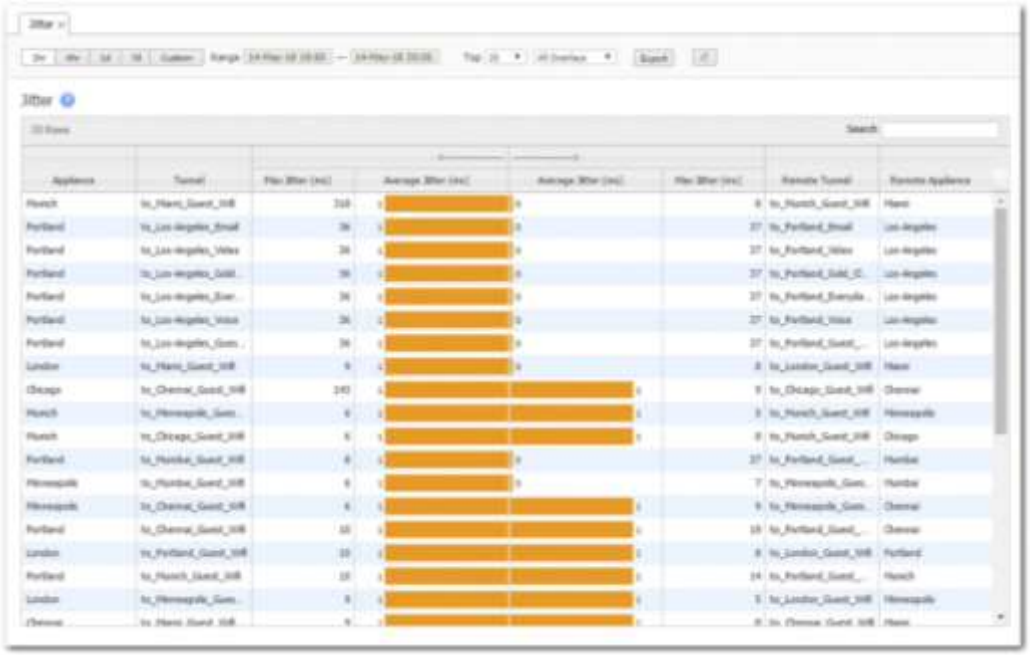
Overview of SD-WAN Prerequisites

This video demonstrates the basic process of creating a Silver Peak tunnel in order to optimize traffic. The video covers tunnel configuration, creating a route policy that specifies what is being optimized, and checking your work via the Current Flows.





Unity Orchestrator uses the TGNM WAN API to target the branches in the network, and associate them to a Transit Gateway, configuring both ends of the tunnel endpoints for each branch as shown below in Figure 2. The EdgeConnect appliance in the branch then establishes standards-based IPSec tunnels that terminate at the head-end gateway in AWS. Orchestrator continuously monitors the status of the connections and redirects traffic to alternate tunnels or gateways as needed.



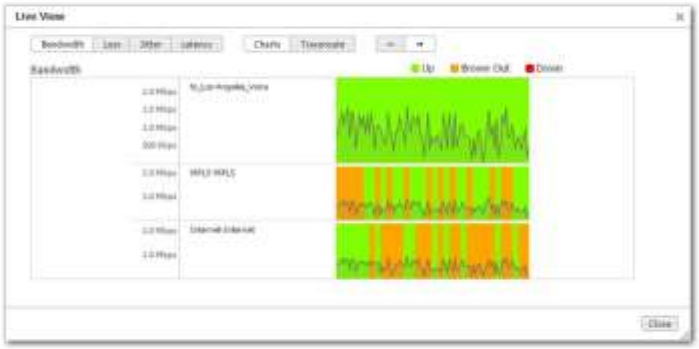
142. The Accused '464 Products practice the step of transmitting quality measurement packets for each of said candidate routes, wherein said quality measurement packets can be used to determine at least one route quality metric. For example, Silver Lake's Unity solution monitors all WAN routes, determining quality metrics that it uses to determine route statistics such as data regarding packet loss, average delay, or average jitter:

Intelligent Internet Breakout

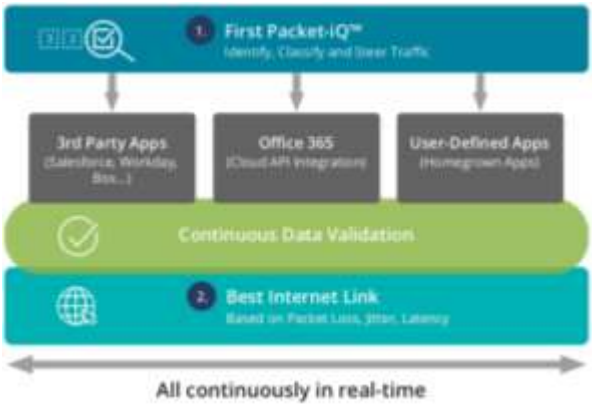
Often customers provision two or more WAN links from remote branch sites to increase network and application availability and performance. These links are used for breaking out traffic locally at each branch. Using the internet as an underlay transport is less expensive than private leased line connections such as MPLS since it offers much higher bandwidth at a given price point. To optimize utilization of the provisioned WAN internet links and to optimize SaaS application performance, EdgeConnect monitors the performance of all links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. EdgeConnect uses statistical learning based on jitter, latency, loss and MOS on all provisioned internet links to dynamically determine which link is performing the best before sending traffic. This optimizes internet break out traffic to deliver the highest SaaS and cloud application performance (see Figure 11). Configuring these policies is fully automated within Silver Peak Unity Orchestrator™ and doesn't require any manual configuration. The Orchestrator also enables configuration of an automated policy for finding the best path for that traffic over the SD-WAN fabric, across MPLS or another WAN service, in the rare case that both underlying internet links are underperforming or are unavailable (see Figure 12).

Live View

Live View shows the live bandwidth, loss, latency, and jitter on all the tunnels. For an overlay, it also shows live tunnel states — **Up**, **Browned Out**, or **Down**.



In real-time, LiveView shows how Silver Peak creates synergy to maintain coverage. The real-time chart shows the SD-WAN overlay at the top and the underlay networks at the bottom. The overlay is green and delivering consistent application performance while both underlays are in persistent brown-out state.

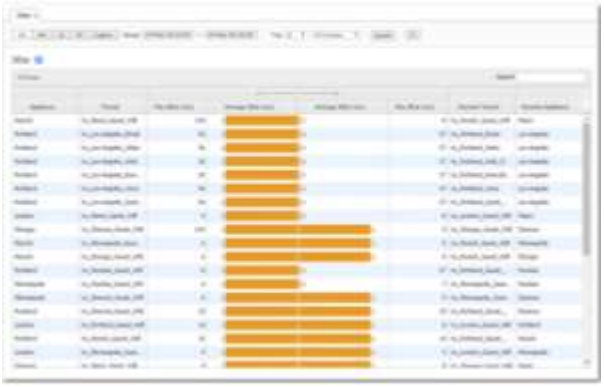


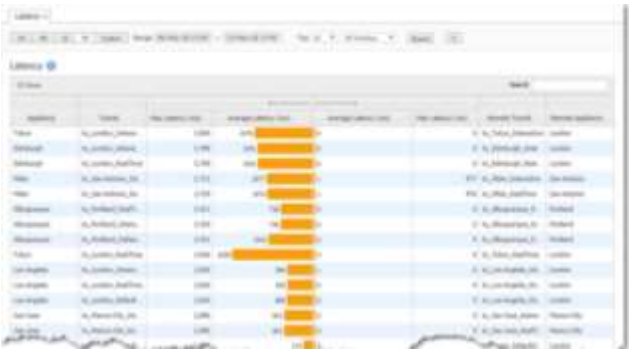
statistical learning to determine the optimal forwarding link, ensuring maximum application performance.

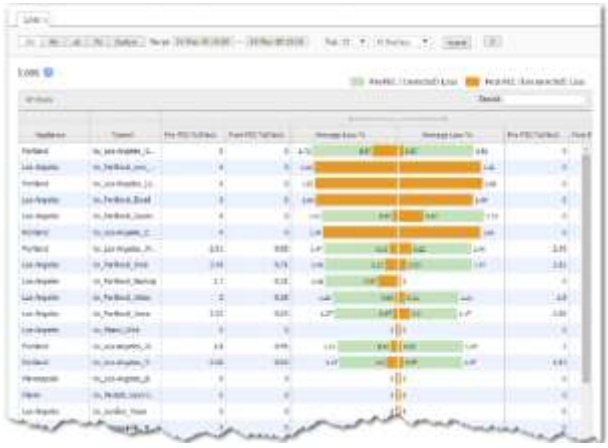
Live View	A live view of the status of your selected tunnel. You can view by bandwidth, loss, jitter, latency, MOS, chart, traceroute, inbound or outbound, and lock the scale.

- Thresholds for **Latency**, **Loss**, or **Jitter** are checked once every second.

143. The Accused ’464 Products practice the step of receiving returned quality measurement packets for each of said candidate routes, wherein said returned quality measurement packets can be used to determine route statistics. For example, based on the monitoring of route quality for all WAN routes, including the receipt of packets, the Accused ’464 Products can derive route statistics, such as data regarding packet loss, average delay, and average jitter:





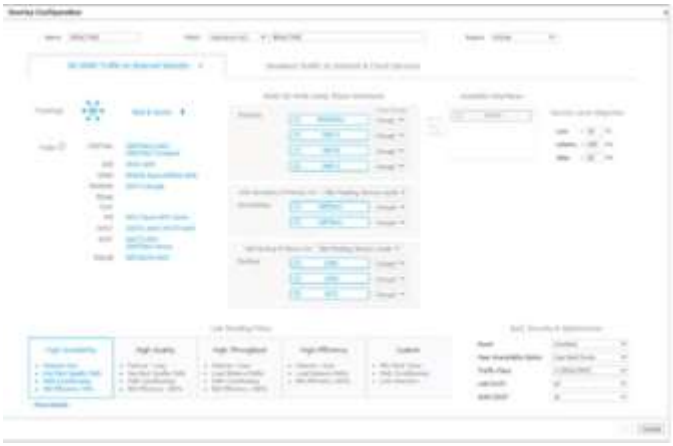


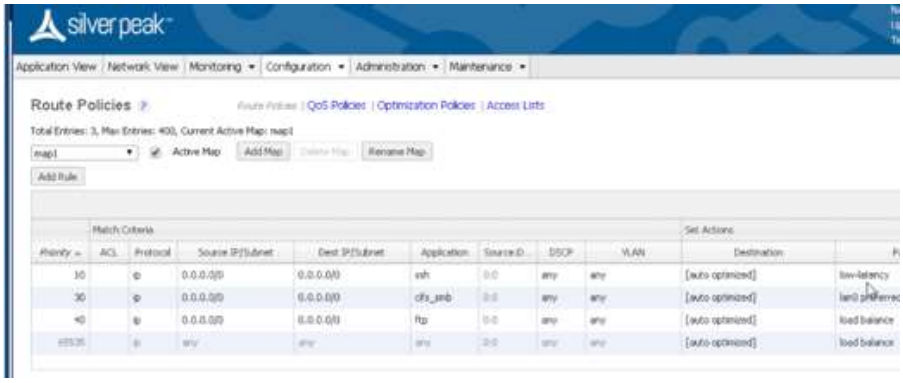
IPsec Encryption Algorithm	For encrypting tunnel data. Choose from **auto**, **AES-256**, or **AES-128**.
Latency	The amount of latency measure in MS.
Loss	The amount of data lost measured in percent.
Jitter	The amount of jitter measured in MS.

144. The Accused ’464 Products practice the step of determining route statistics, wherein said route statistics are based on routing information contained within said quality measurement packets. For example, as noted above, the Accused ’464 Products determine route statistics, such as data regarding packet loss, average delay, and average jitter, based on measurements of packets sent across all routes, and such statistics can be visualized in various ways, such as with graphs and diagrams:

145. The Accused ’464 Products practice the step of configuring a route ordering schedule based on user set levels of route characteristics. For example, the Accused ’464 Products

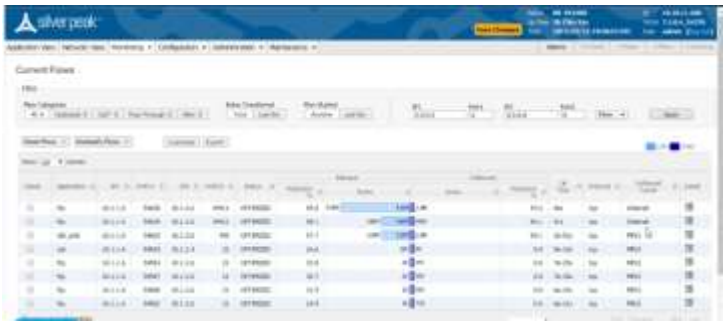
enable the configuration of policies based on business priorities that will be enforced in route ordering and scheduling of packets:





146. The Accused ’464 Products practice the step of scoring each of said candidate routes based on route statistics and said route ordering schedule, wherein a scoring table is

configured that includes a quality score and one or more of packet loss, average delay, and average jitter. For example, the Accused '464 Products score each route based upon the determined route statistics, and display the scores and rankings in, *e.g.*, tables, charts, and graphs, along with a quality score called mean opinion score (MOS) of quality:



- Thresholds for **Latency**, **Loss**, or **Jitter** are checked once every second.
  - Receiving 3 successive measurements in a row that exceed the threshold puts the tunnel into a brownout situation and flows will attempt to fail over to another tunnel within the next 100mS.
  - Receiving 3 successive measurements in a row that drop below the threshold will drop the tunnel out of brownout.

Often customers provision two or more WAN links from remote branch sites to increase network and application availability and performance. To optimize utilization of the provisioned WAN internet links, EdgeConnect monitors the performance of the links in real-time by continuously measuring packet loss, jitter, latency, and mean opinion score (MOS). EdgeConnect uses statistical learning to dynamically determine the optimal link for breaking out traffic, thus maintaining peak application performance. This optimizes internet break out traffic to deliver the highest SaaS and cloud application performance (see Figure 1). Configuring these policies is fully automated within the Silver Peak Unity Orchestrator™ management interface and doesn't require any manual configuration. Orchestrator also enables configuration of an automated policy for finding the best path for that traffic over the SD-WAN fabric, across MPLS or another WAN service, in the rare case that both underlying internet links are underperforming or unavailable (see Figure 2).

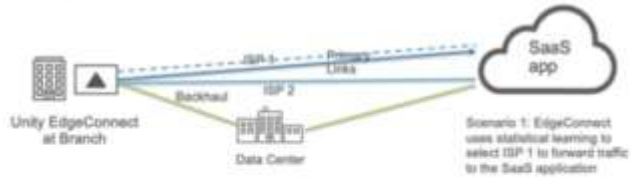


Figure 1: To optimize utilization of the provisioned WAN internet links (ISP 1 and ISP 2), EdgeConnect monitors the performance of the two links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. Using statistical learning, EdgeConnect dynamically select ISP 1 to send traffic to the SaaS application.

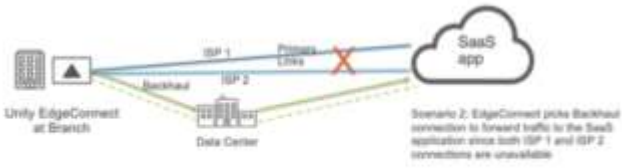


Figure 2: If both ISP 1 and ISP 2 connections become unavailable, EdgeConnect automatically chooses the configured backup transport service that backhauls traffic through the data center.

Highest End-user Quality of Experience

Selecting the best path to direct packets eliminates any additional latency experienced by your SaaS applications and IaaS workloads. This delivers the highest quality of experience to users and results in happy customers, prospects, and employees. In addition, intelligent internet breakout can also be used to automatically address brownout or blackout conditions on any link. For instance, if a branch is served by one MPLS connection and one internet connection, before sending any packets over the internet connection, the EdgeConnect appliance confirms the connection quality. If for some reason, the internet connection is experiencing loss, latency, jitter or MOS greater than a pre-configured threshold, the EdgeConnect appliance will automatically select the MPLS connection to send packets. This ensures that no matter what happens, enterprise applications, whether hosted in the data center, hosted in IaaS, UCaaS, or SaaS, always operate at peak performance. The new Silver Peak intelligent internet breakout feature dramatically improves business productivity while enabling businesses to increase efficiency.

Brownout Thresholds specify the triggers for switching from a **Primary** service to a **Backup** service.

- Traffic is routed to the Primary service unless a threshold for Loss, Latency, or Jitter has been exceeded.
- When **Blackout** is selected, then the Backup service is used only if the Primary service goes down completely.
- When **Brownout** is selected, then the Backup service is used until each aspect of the Primary service is again within normal limits.

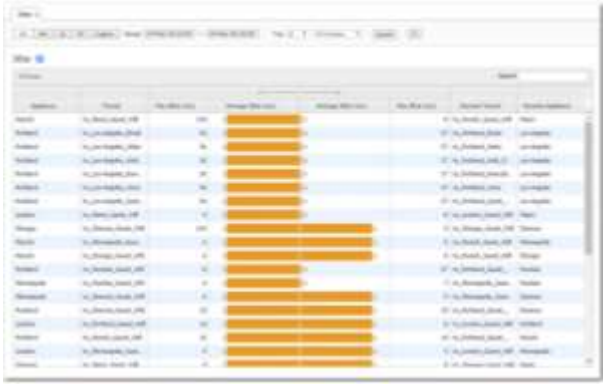
A comprehensive dashboard (Figure 1) provides a complete at-a-glance view along with customizable widgets to monitor network attributes and applications in real time. IT defined widgets provide granular details on SD-WAN appliances, including their location, active tunnels, logical topology, appliance health heatmap, top talkers, alarms, mean opinion score (MOS), applications and domains accessed, bandwidth consumed, flow count, latency, jitter, and packet loss (Figure 2).

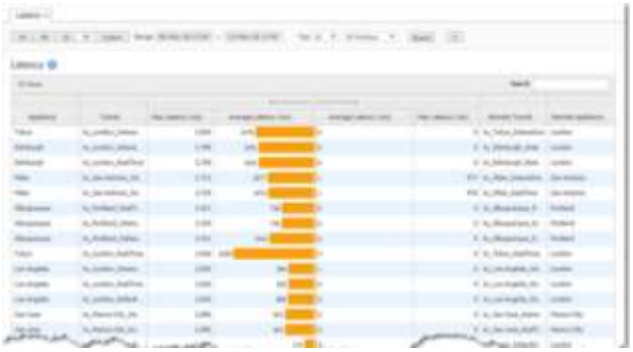
You may also want to create a Route Policy entry when multiple tunnels exist to the remote **peer**, and you want the appliance to dynamically select the best path based on one of these criteria:

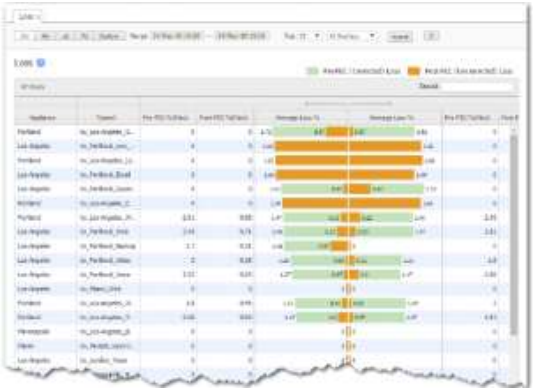
- load balancing
- lowest loss
- lowest latency
- specified tunnel

Intelligent Internet Breakout

Often customers provision two or more WAN links from remote branch sites to increase network and application availability and performance. These links are used for breaking out traffic locally at each branch. Using the internet as an underlay transport is less expensive than private leased line connections such as MPLS since it offers much higher bandwidth at a given price point. To optimize utilization of the provisioned WAN internet links and to optimize SaaS application performance, EdgeConnect monitors the performance of all links by continuously measuring the packet loss, jitter, latency and mean opinion score (MOS) in real-time. EdgeConnect uses statistical learning based on jitter, latency, loss and MOS on all provisioned internet links to dynamically determine which link is performing the best before sending traffic. This optimizes internet break out traffic to deliver the highest SaaS and cloud application performance (see Figure 11). Configuring these policies is fully automated within Silver Peak Unity Orchestrator™ and doesn't require any manual configuration. The Orchestrator also enables configuration of an automated policy for finding the best path for that traffic over the SD-WAN fabric, across MPLS or another WAN service, in the rare case that both underlying internet links are underperforming or are unavailable (see Figure 12).

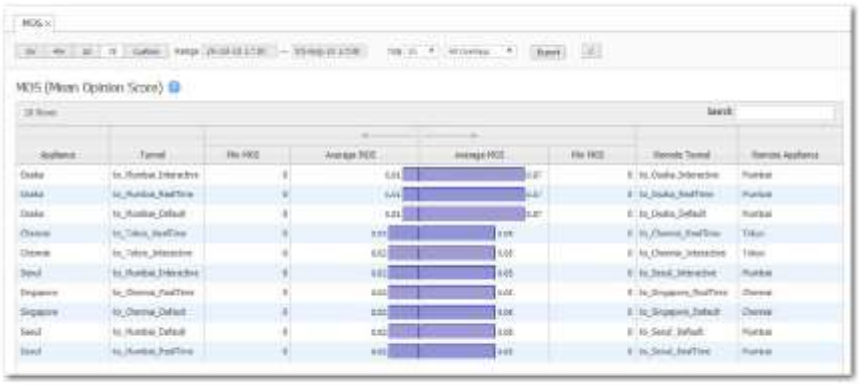






Mean Opinion Score (MOS) - Summary

The Mean Opinion Score (MOS) is a commonly used measure for video, audio, and audiovisual quality evaluation. Perceived quality is rated on a theoretical scale of 1 to 5; the higher the number, the better the quality.



The value can be affected by loss, latency, and jitter. In practice, a value of **4.4** is considered an excellent quality target.

147. Additionally, HPE has been, and currently is, an active inducer of infringement of the ’464 patent under 35 U.S.C. § 271(b) and contributory infringement of the ’464 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

148. HPE has actively induced, and continues to actively induce, infringement of the ’464 patent by intending that others use, offer for sale, sell and/or import products and/or services covered by the ’464 patent, including but not limited to HPE Unity EdgeConnect, and any HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States the foregoing products and/or services that directly infringe the ’464 patent as described above.

149. HPE has contributed to, and continues to contribute to, the infringement of the ’464 patent by others by offering to sell, selling, or otherwise commercially offering products and/or

services that, when installed and configured result in a system as intended by HPE, that directly infringes the ’464 patent.

150. HPE knew of the ’464 patent, or should have known of the ’464 patent, but was willfully blind to its existence. Upon information and belief, HPE has had actual knowledge of the ’464 patent since at least as early as the service upon HPE of this Complaint. On information and belief, HPE had actual knowledge of the ’464 patent in and around March 25, 2020, on information and belief, HPE was aware of IV’s allegations against VMware, Inc. relating to infringement of the ’464 patent by VMware’s methods for accessing network routes for use in establishing a network communications link. Additionally, HPE was aware of the ’464 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the ’464 patent.

151. HPE has committed, and continues to commit, affirmative acts that cause infringement of the ’464 patent with knowledge of the ’464 patent and knowledge or willful blindness that the induced acts constitute infringement of the ’464 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customary way as desired and intended by HPE, infringe the ’464 patent and/or by directly or indirectly providing instructions on how to use its products and/or services in a manner or configuration that infringes the ’464 patent, including those found at the following:

- https://www.silver-peak.com/resource-center/unity-edge-connect-data-sheet-solution
- https://www.silver-peak.com/documentation/orchestrator-user-guide
- https://www.silver-peak.com/resource-center/how-create-tunnel

- https://www.silver-peak.com/resource-center/sd-wan-dynamic-path-control-demo
- https://www.silver-peak.com/resource-center/solution-briefs/silver-peak-sd-wan-and-aws-transit-gateway-network-manager
- https://www.silver-peak.com/sites/default/files/infoctr/silver-peak-datasheet-unity_edgeconnect-sd-wan-solution-service-provider.pdf
- https://www.silver-peak.com/sites/default/files/infoctr/silver-peak_ss_voip.pdf
- https://blog.silver-peak.com/modern-cloud-first-enterprises-require-intelligent-internet-breakout

152. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that when used cause the direct infringement of the ’464 patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such as software, that are especially made or especially adapted for use in infringement of the ’464 patent and are not staple articles or commodities of commerce suitable for substantial non-infringing use.

153. As a result of HPE’s acts of infringement, IV has suffered and will continue to suffer damages in an amount to be proved at trial.

COUNT VI
(HPE’s Infringement of U.S. Patent No. 8,023,991)

154. Paragraphs 1-153 are incorporated by reference as if fully set forth herein.

155. The inventions claimed by the ’991 patent, taken alone or in combination, were not well-understood, routine or conventional to one of ordinary skill in the art at the time of the invention of the ’991 patent. Rather, the patent teaches and claims an improved way to optimize wireless network performance particularly in crowded and dense wireless environments with lots of user devices (stations) and network access points (APs) competing for (sharing) limited resources. The inventions improved upon then-existing techniques for the configuration and

management of wireless networks, which often required manual site engineering to determine and control the placement of APs in a given area and lacked the ability to efficiently and effectively share the same frequency channel with other nearby APs, resulting in interference and sub-optimal resource usage. For example, the inventions improved prior art wireless communications environments and systems by using an approach that enabled an AP transmitting on a given frequency channel to perform automatic power adjustments of transmit power levels, with the power adjustments being based on received indications of the transmit power levels being used by other APs, and the power adjustments being set to reduce interference with other APs that are sharing the same frequency channel.

156. The inventions represented a technical solution to an unsolved technological problem. The specification of the '991 patent describes, in technical detail, each of the limitations in the claim, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claim differs markedly from what had been conventional or generic in the industry at the time of the invention of the '991 patent. More specifically, the'991 patent is directed to detecting when two APs are using the same radio frequency channel (co-channel APs), receiving messages from the co-channel APs, and maintaining indications of the transmit power level of other APs in the area including the co-channel APs, and, responsive to this information about the transmit power levels of the various APs, instructing one of the co-channel APs to adjust its transmit power to decrease interference with the other co-channel AP.

157. The computer program product covered by the asserted claim therefore differs markedly from the conventional and generic systems in use at the time of this invention, which,

inter alia, lacked the above noted combination of detecting when multiple APs are using the same channel, and instructing one of those APs to adjust its power level so as to decrease interference with the other AP on that same channel.

158. As described above, the ’991 patent is drawn to solving a specific, technical problem arising in the context of wireless network management and configuration. Consistent with the problem addressed being rooted in such wireless network communications technology, the solutions provided by the ’991 patent naturally are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

159. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, claim 1 of the ’991 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the claim of the ’991 patent. HPE’s products and/or services that infringe the ’991 patent include, but are not limited to, the Adaptive Radio Management (ARM) feature of HPE-owned Aruba’s operating system, ArubaOS (and all versions/editions that support the ARM feature), including the Aruba 7200 Series Mobility Controllers on which ArubaOS runs; the Mobility Master feature, particularly the AirMatch Workflow functionality, of ArubaOS 8 (and all versions/editions that support such functionality); and any other of HPE’s products and/or services, either alone or in combination, that operate in substantially the same manner (together the “Accused ’991 Products”).

Claim 1 of the ’991 patent is reproduced below:

> *1. A computer program product recorded on a computer-readable medium, comprising:*
> *logic for detecting that a first access point is using a radio frequency channel;*
> *logic for detecting that a second access point is also using the radio frequency channel, the detecting logic including:*
> *logic for receiving messages from the second access point; and*

logic for maintaining indications of the transmit power levels of other access points including the second access point; and

logic, responsive to the indications of the transmit power levels of other access points maintained by the detecting logic, for instructing the first access point to adjust transmit power to decrease interference with the second access point detected to be using the radio frequency channel; wherein the first access point adjusts transmit power as instructed.

160. The Accused '991 Products comprise a computer program product recorded on a computer-readable medium. For example, Aruba's ARM functionality is a distributed approach to enable self-configuring, self-healing wireless networks, and it includes a controller that computes the most optimized setting and instructs access points (APs) to work accordingly. ARM is part of the base ArubaOS operating system, which is a computer program product recorded on a computer-readable medium, and which is available on all Aruba Mobility Controllers and APs. Aruba lists among the minimum system requirements for ArubaOS at least 8GB RAM, 4GB Memory, and 10GB Disk Space. Similarly, for example, Aruba's Mobility Master with AirMatch functionality also requires and is stored on RAM, Memory, and Disk Space, constituting a computer program product recorded on a computer-readable medium. AirMatch offers similar benefits but improves upon and goes beyond ARM by utilizing AI / machine learning to provide automated RF optimization for wireless networks, including *inter alia* automated channel optimization and transmit power adjustment:

Aruba's ARM is a distributed approach to enable self configuring, self healing wireless networks. In order to fully utilize the available spectrum, increase the system capacity, and the number of users supported, ARM dynamically learns about the RF medium and adapts accordingly. This is accomplished by AP's that periodically scan other channels, analyzes the interference level seen on other channels and reports it back to the controller.

The controller then computes the most optimized setting and instructs the AP to work accordingly. There is a small amount of time when the AP's leave their channel to conduct scanning operations that are essential to ARM. Where there are real time applications such as video/voice running on the network, this periodic scan might result in latency, jitter and eventually degrade the quality of the signal.

System requirements

Listed below are the minimum hypervisor host system requirements for ArubaOS to run as a guest Virtual Machine (VM) and the resources required for the VM to be functional:

- Quad-core Core i5 1.9 GHz processors with hyper threading enabled.
- 8GB RAM
- Minimum 2 physical NICs on the ESXi host

ArubaOS VM requirement

Listed below are the minimum resources required for the ArubaOS VM to function:

- 3 vCPUs.
- 4GB Memory.
- 10GB Disk space.
- 4 Virtual NICs.

Aruba uses information gathered from APs and air monitors (AMs) that scan the RF environment to provide information to the ARM algorithms and services. The infrastructure has a network-wide view of APs and clients, and this information is used to optimize the network and to provide an enhanced client experience. ARM is a part of the base ArubaOS™ and is available on all Aruba Mobility Controllers and APs.

The Aruba Mobility Master provides a 64-bit virtualized software-based managed platform on VirtualMachine (VM) architecture.

Mobility Master is the centralized management platform for deployment in a virtualized network infrastructure. The Mobility Master operates on VM platforms in VMware environments and can reside withother virtualized appliances.

Aruba AirMatch goes beyond Adaptive Radio Management (ARM) by utilizing AI/machine learning to provide automated radio frequency (RF) optimization. Instead of looking at each individual AP like in the ARM model, AirMatch looks at analytics across the entire WLAN.

AirMatch is a key component of Aruba's AI-powered Mobility solution and is supported in environments utilizing the Aruba Mobility Master (ArubaOS 8+). This delivers automated system-wide channel, bandwidth and EIRP optimization – no manual intervention required.

THE FOLLOWING TECHNOLOGIES IN ARUBAOS 8 ARE ONLY SUPPORTED IN THE MOBILITY MASTER	
Feature	**Benefit**
AirMatch	Aruba further enhances the Adaptive Radio Management (ARM) technology with AirMatch – the new automated channel optimization, transmit power adjustment and channel width tuning system that utilizes dynamic machine learning intelligence to automatically generate the optimal view of the entire WLAN network.

AirMatch Benefits	
Even Channel assignment	Provides even distribution of radios across available channels, interference mitigation and maximized system capacity.
Dynamic Channel width adjustment	Dynamically adjusts between 20MHz, 40MHz and 80MHz to match the density of your environment.
Automatic transmit power adjustment	Examines the entire WLAN coverage and automatically adjusts the transmit power of APs to ensure the best coverage and user experience.

161. The Accused '991 Products comprise a computer program product recorded on a computer-readable medium including logic for detecting that a first access point is using a radio frequency channel. For example, Aruba offers configuration settings enabling detection of APs and the frequency channels to which they are assigned or on which they are operating, for both its ARM and AirMatch products:

ARM Coverage and Interference Metrics

ARM computes coverage and interference metrics for each valid channel, and chooses the best performing channel and transmit power settings for each AP's RF environment. Each AP gathers other metrics on their ARM-assigned channel to provide a snapshot of the current RF health state.

Channel or Power Assignment

The channel or power assignment feature automatically assigns channel and power settings for all the Instant APs in the network according to changes in the RF environment. This feature automates many setup tasks during network installation and the ongoing operations when RF conditions change.

AI-POWERED INNOVATION

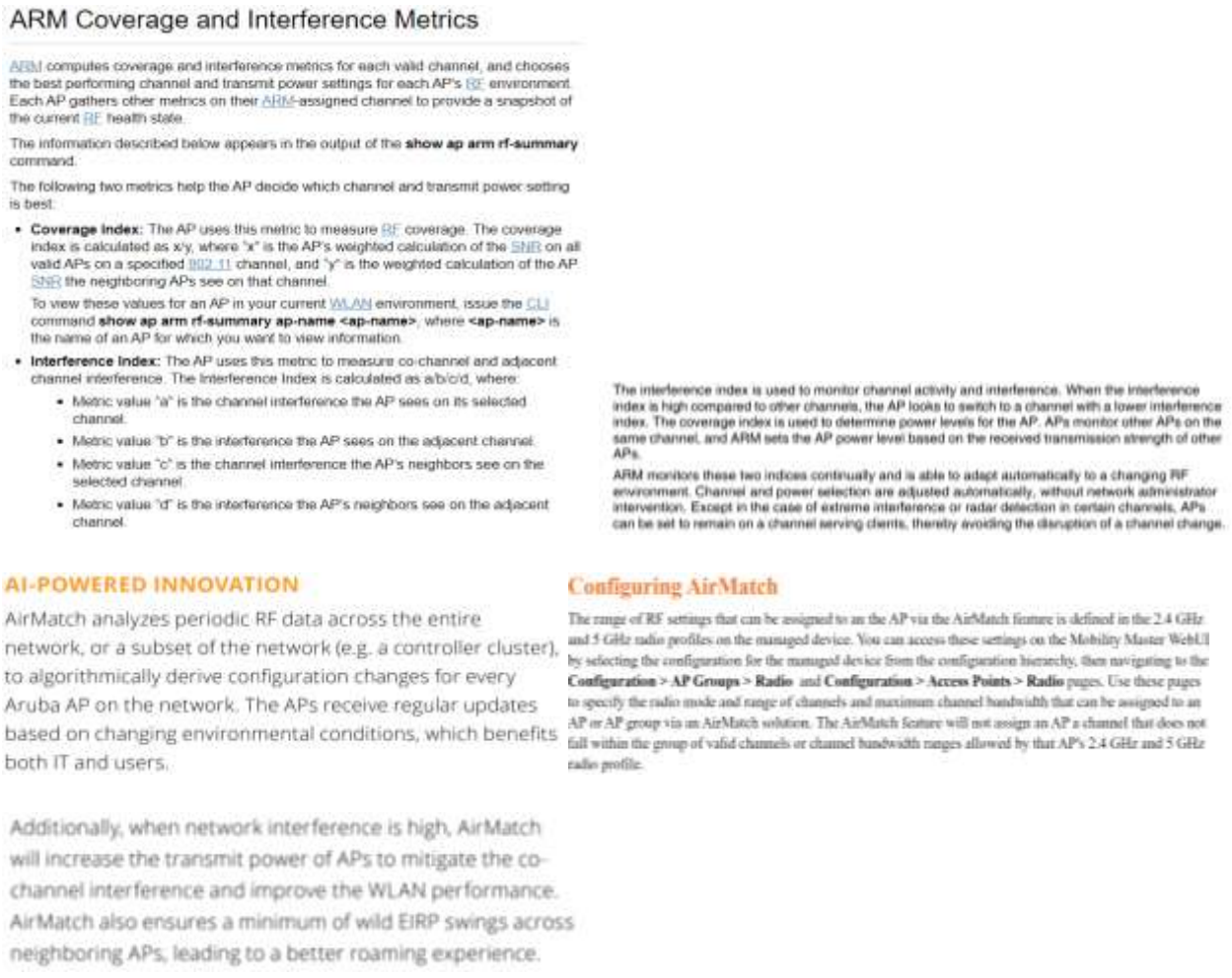
AirMatch analyzes periodic RF data across the entire network, or a subset of the network (e.g. a controller cluster), to algorithmically derive configuration changes for every Aruba AP on the network. The APs receive regular updates based on changing environmental conditions, which benefits both IT and users.

Configuring AirMatch

The range of RF settings that can be assigned to an the AP via the AirMatch feature is defined in the 2.4 GHz and 5 GHz radio profiles on the managed device. You can access these settings on the Mobility Master WebUI by selecting the configuration for the managed device from the configuration hierarchy, then navigating to the **Configuration > AP Groups > Radio** and **Configuration > Access Points > Radio** pages. Use these pages to specify the radio mode and range of channels and maximum channel bandwidth that can be assigned to an AP or AP group via an AirMatch solution. The AirMatch feature will not assign an AP a channel that does not fall within the group of valid channels or channel bandwidth ranges allowed by that AP's 2.4 GHz and 5 GHz radio profile.

162. The Accused '991 Products comprise a computer program product recorded on a computer-readable medium including logic for detecting that a second access point is also using the radio frequency channel. Simply put, APs monitor other APs on the same channel and thus ARM contains logic for detecting that a first and second AP are using the same channel. For

example, ARM uses various metrics to help APs decide which channel and transmit power setting is best, including calculating the signal-to-noise ratio (SNR) for all valid APs on a particular 802.11 frequency channel as well as considering a weighted calculation of the SNR the neighboring APs see on that same frequency channel. Similarly, for example, ARM takes into account and interference index with values including the interference an AP sees on its selected channel as well as the interference neighboring APs see on that same selected channel. Also, for example, AirMatch can analyze RF data across the entire network, including all APs on the network, and algorithmically derive configuration changes for every AP on the network:



ARM Coverage and Interference Metrics

ARM computes coverage and interference metrics for each valid channel, and chooses the best performing channel and transmit power settings for each AP's RF environment. Each AP gathers other metrics on their ARM-assigned channel to provide a snapshot of the current RF health state.

The information described below appears in the output of the **show ap arm rf-summary** command.

The following two metrics help the AP decide which channel and transmit power setting is best:

- **Coverage Index:** The AP uses this metric to measure RF coverage. The coverage index is calculated as x/y, where "x" is the AP's weighted calculation of the SNR on all valid APs on a specified 802.11 channel, and "y" is the weighted calculation of the AP SNR the neighboring APs see on that channel.

  To view these values for an AP in your current WLAN environment, issue the CLI command **show ap arm rf-summary ap-name <ap-name>**, where **<ap-name>** is the name of an AP for which you want to view information.
- **Interference Index:** The AP uses this metric to measure co-channel and adjacent channel interference. The Interference Index is calculated as a/b/c/d, where:
  - Metric value "a" is the channel interference the AP sees on its selected channel.
  - Metric value "b" is the interference the AP sees on the adjacent channel.
  - Metric value "c" is the channel interference the AP's neighbors see on the selected channel.
  - Metric value "d" is the interference the AP's neighbors see on the adjacent channel.

The interference index is used to monitor channel activity and interference. When the interference index is high compared to other channels, the AP looks to switch to a channel with a lower interference index. The coverage index is used to determine power levels for the AP. APs monitor other APs on the same channel, and ARM sets the AP power level based on the received transmission strength of other APs.

ARM monitors these two indices continually and is able to adapt automatically to a changing RF environment. Channel and power selection are adjusted automatically, without network administrator intervention. Except in the case of extreme interference or radar detection in certain channels, APs can be set to remain on a channel serving clients, thereby avoiding the disruption of a channel change.

AI-POWERED INNOVATION

AirMatch analyzes periodic RF data across the entire network, or a subset of the network (e.g. a controller cluster), to algorithmically derive configuration changes for every Aruba AP on the network. The APs receive regular updates based on changing environmental conditions, which benefits both IT and users.

Additionally, when network interference is high, AirMatch will increase the transmit power of APs to mitigate the co-channel interference and improve the WLAN performance. AirMatch also ensures a minimum of wild EIRP swings across neighboring APs, leading to a better roaming experience.

Configuring AirMatch

The range of RF settings that can be assigned to an the AP via the AirMatch feature is defined in the 2.4 GHz and 5 GHz radio profiles on the managed device. You can access these settings on the Mobility Master WebUI by selecting the configuration for the managed device from the configuration hierarchy, then navigating to the **Configuration > AP Groups > Radio** and **Configuration > Access Points > Radio** pages. Use these pages to specify the radio mode and range of channels and maximum channel bandwidth that can be assigned to an AP or AP group via an AirMatch solution. The AirMatch feature will not assign an AP a channel that does not fall within the group of valid channels or channel bandwidth ranges allowed by that AP's 2.4 GHz and 5 GHz radio profile.

163. The Accused '991 Products comprise a computer program product recorded on a computer-readable medium including detecting logic that further includes logic for receiving

messages from the second access point. For example, as part of the ARM functionality, APs periodically scan other channels and analyze the interference level seen on them and report the results back to the controller to enable it to dynamically learn about the RF medium and adapt accordingly. The Accused '991 Products also include an Over the Air Updates feature that allows an AP to get information about its RF environment from its neighboring APs, such as when an AP scans a foreign (non-home) channel and sends an Over-the-Air update in a particular 802.11 frame that contains information about that AP's home channel, the current transmission EIRP value of the home channel, and one-hop neighboring APs seen by that AP. Similarly, for example, as part of AirMatch, each AP in an Aruba Mobility Master deployment measures its RF environment periodically and sends messages about the radio feasibility to the managed device based on that AP's hardware capability and other factors. The managed device forwards these messages to the Mobility Master. Also, for example, when Mobility Master first detects APs on the network it enters an initial optimization phase and collects data from all the APs and generates configurations for the APs. Thus, the Accused '991 Products contain logic for receiving such messages back from the APs, including the second AP:

Aruba's ARM is a distributed approach to enable self configuring, self healing wireless networks. In order to fully utilize the available spectrum, increase the system capacity, and the number of users supported, ARM dynamically learns about the RF medium and adapts accordingly. This is accomplished by AP's that periodically scan other channels, analyzes the interference level seen on other channels and reports it back to the controller.

The controller then computes the most optimized setting and instructs the AP to work accordingly. There is a small amount of time when the AP's leave their channel to conduct scanning operations that are essential to ARM. Where there are real time applications such as video/voice running on the network, this periodic scan might result in latency, jitter and eventually degrade the quality of the signal.

ARM Monitoring and Management

When ARM is enabled, the Aruba AP dynamically scans all 802.11 channels in its regulatory domain at regular intervals and will report everything it sees to the controller on each channel it scans (by default, 802.11n-capable APs scan channels in all regulatory domains). This includes, but is not limited to, data regarding WLAN coverage, interference, and intrusion detection. You can retrieve this information from the controller

Introduction : The Over the Air Updates feature allows an AP to get information about its RF environment from its neighbors, even the AP cannot scan. If this feature is enabled, when an AP on the network scans a foreign (non-home) channel, it sends an Over-the-Air (OTA) update in an 802.11 management frame that contains information about that AP's home channel, the current transmission EIRP value of the home channel, and one-hop neighbors seen by that AP.

AirMatch Channel Assignments

Each AP in a Mobility Master deployment measures its RF environment for a five minute period, every 30 minutes by default. The AP then sends AMON messages about the radio feasibility to the managed device based on that AP's hardware capability, radio and regulatory domain, and RF neighbors. The managed device forwards these messages to the Mobility Master. The Mobility Master adds this information to a database, computes an optimal solution, and deploys the latest RF plan by sending updated settings to the APs. By default, this configuration update is sent at 5 AM (as per the Mobility Master system clock), but time of this configuration update can be modified via the AirMatch profile.

Initial RF Calculations

The database for the AirMatch service is empty when Mobility Master first boots up. When Mobility Master first detects APs on the network, it enters its initial optimization phase, collects data from all the APs, and generates an incremental solution every 30 minutes (by default) for the next eight hours. When this initial eight-hour period has elapsed, the AirMatch service will periodically calculate a new RF configuration for these devices.

164. The Accused '991 Products comprise a computer program product recorded on a computer-readable medium including detecting logic that further includes logic for maintaining indications of the transmit power levels of other access points including the second access point. For example, as noted above, ARM uses various metrics and gathers and maintains certain data, including calculating the signal-to-noise ratio (SNR) for all valid APs on a particular 802.11 frequency channel as well as considering a weighted calculation of the SNR the neighboring APs see on that same frequency channel. Similarly, for example, ARM takes into account and interference index with values including the interference an AP sees on its selected channel as well as the interference neighboring APs see on that same selected channel. These indexes are used to monitor channel activity and interference and determine power levels for APs. Furthermore, in Aruba's ARM system, APs monitor their local environment for interference, noise, and signals being received from other Aruba APs, and repots this information back to the controller. Summaries of RF data such as interference and transmit power of various access points are maintained by ARM. Also, for example, with AirMatch, APs send messages about the radio feasibility to the managed device based on each AP's hardware capability, radio and regulatory domain, and RF neighbors, and the managed device forwards these messages to the Mobility

Master, which adds that information to a database. As part of transmit power adjustment, AirMatch examines the entire WLAN coverage. For non-static channels, AirMatch selects a channel with a minimum interference index from the channels without high noise or a radar condition:

- **Coverage Index**: The AP uses this metric to measure RF coverage. The coverage index is calculated as x/y, where "x" is the AP's weighted calculation of the SNR on all valid APs on a specified 802.11 channel, and "y" is the weighted calculation of the AP SNR the neighboring APs see on that channel.

  To view these values for an AP in your current WLAN environment, issue the CLI command **show ap arm rf-summary ap-name <ap-name>**, where **<ap-name>** is the name of an AP for which you want to view information.
- **Interference Index**: The AP uses this metric to measure co-channel and adjacent channel interference. The Interference Index is calculated as a/b/c/d, where:
  - Metric value "a" is the channel interference the AP sees on its selected channel.
  - Metric value "b" is the interference the AP sees on the adjacent channel.
  - Metric value "c" is the channel interference the AP's neighbors see on the selected channel.
  - Metric value "d" is the interference the AP's neighbors see on the adjacent channel.

Aruba's ARM technology takes the guesswork out of AP deployments. Once APs are brought up, they immediately begin monitoring their local environment for interference, noise, and signals being received from other Aruba APs. This includes detection of other APs or Wi-Fi networks and the channels they are using.

This information is reported back to the controller, which is then able to control the optimal channel assignment and power levels for each AP in the network – even where 802.11ac has been deployed with mixed VHT20, VHT40 and VHT80 channel types.

The interference index is used to monitor channel activity and interference. When the interference index is high compared to other channels, the AP looks to switch to a channel with a lower interference index. The coverage index is used to determine power levels for the AP. APs monitor other APs on the same channel, and ARM sets the AP power level based on the received transmission strength of other APs.

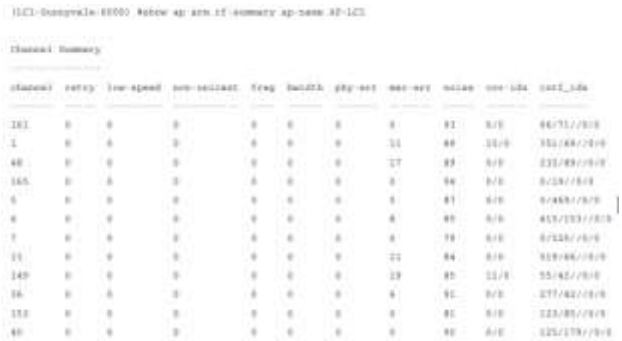
ARM monitors these two indices continually and is able to adapt automatically to a changing RF environment. Channel and power selection are adjusted automatically, without network administrator intervention. Except in the case of extreme interference or radar detection in certain channels, APs can be set to remain on a channel serving clients, thereby avoiding the disruption of a channel change.

ARM Monitoring and Management

When ARM is enabled, the Aruba AP dynamically scans all 802.11 channels in its regulatory domain at regular intervals and will report everything it sees to the controller on each channel it scans (by default, 802.11n-capable APs scan channels in all regulatory domains). This includes, but is not limited to, data regarding WLAN coverage, interference, and intrusion detection. You can retrieve this information from the controller

Table 10 ARM RF Summary Description (Continued)

Column Heading	Description
Phy-err	The amount of frames with physical errors seen on a channel (%).
Mac-err	The amount of frames with MAC errors seen on a channel (%).
Noise	The APs noise floor (0 dBm).
Cov-idx	The amount of RF coverage from valid Aruba APs on a specific channel (#).
Intf_idx	An 802.11 RF interference summary that is categorized into four values: • Single channel interference is calculated by the AP • Interference from APs on adjacent overlapping channels is calculated by the AP • Single channel interference reported by neighboring APs • Interference on adjacent overlapping channels is reported by neighboring APs



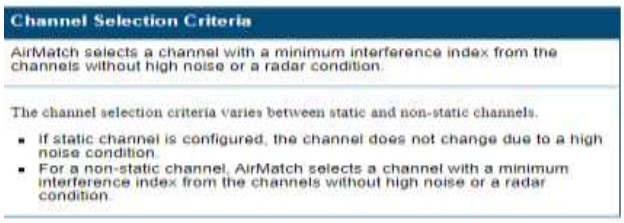
Transmit power adjustment

AirMatch examines the entire WLAN coverage and automatically adjusts the transmit power of APs to ensure the best coverage and user experience. For example, when an AP is down, it creates a coverage hole in the network. AirMatch will increase the transmit power of neighboring APs to extend the coverage. As shown in Figure 1 and Figure 2, in order to extend coverage for the gap in the center of the rectangular area, AirMatch adjusted the EIRP values for all of the APs around the hole to 9dbm for 2.4GHz and 16dbm for 5GHz symmetrically; while ARM adjusted the APs' EIRP values asymmetrically due to its local view of the network.

AirMatch Channel Assignments

Each AP in a Mobility Master deployment measures its RF environment for a five minute period, every 30 minutes by default. The AP then sends AMON messages about the radio feasibility to the managed device based on that AP's hardware capability, radio and regulatory domain, and RF neighbors. The managed device forwards these messages to the Mobility Master. The Mobility Master adds this information to a database, computes an optimal solution, and deploys the latest RF plan by sending updated settings to the APs. By default, this configuration update is sent at 5 AM (as per the Mobility Master system clock), but time of this configuration update can be modified via the AirMatch profile.

Figure 70: *Links to the Radio Statistics page on APs/Devices > Monitoring for an AP*



165. The Accused '991 Products comprise computer programs recorded on a computer-readable medium including logic, responsive to the indications of the transmit power levels of other access points maintained by the detecting logic, for instructing the first access point to adjust transmit power to decrease interference with the second access point detected to be using the radio frequency channel; wherein the first access point adjusts transmit power as instructed. For example, the ARM system can detect co-channel interference which frequently occurs when APs are densely deployed near each other. The ARM algorithm takes account of the overlap in coverage causing the co-channel interference and may instruct APs to reduce transmit power on APs that are interfering with one another. ARM's power assignment feature automatically assigns power setting for all APs in a given network according to changes in the RF environment and strives to mitigate co-channel interference by, *inter alia*, performing automatic power assignments to maximize capacity across the network. Additionally, for example, ARM instructs APs at what levels to set their minimum and maximum transmit power they are permitted to use, and if APs detect strong signals from other APs on the same channel, they may decrease their power levels accordingly. ARM uses indexes to calculate the optimum channel and transmit power for each AP, computing these most optimized settings and instructing the AP to work accordingly. ARM can be enabled to perform dynamic channel selection and output power assignment to mitigate co-channel interference. Also, for example, in AirMatch's Mobility Master system, each AP measures its RF environment periodically and repeatedly, and Mobility Master uses this information to compute an optimal solution and deploys the latest RF plan, including indications

of transmit powers that should be used by the APs, by sending the information in that plan to the APs. When network interference is high, AirMatch can increase the transmit power of APs to mitigate co-channel interference:

Co-channel interference often occurs when access points (APs) are densely deployed. When the RF coverage begins to overlap, instead of increased coverage, transmission interference occurs. This interference is more common in the 2.4 GHz (802.11 b/g) band where there are typically only three channels available, and the radio transmissions propagate further. Dual-mode APs are typically deployed to support transmission in the 5 GHz band where 23 channels are available and transmission propagation is reduced.

In a dense deployment, the ARM algorithm notices this overlap in coverage and continues to reduce power and adjust channels on APs that are interfering with one another to balance the network. Even with this tuning in very dense deployments, the network the RF environment could still experience interference, which provides a suboptimal user experience. After tuning the APs, the system can use two methods to mitigate this interference.

Aruba Adaptive Radio Management (ARM), built into ArubaOS, dynamically adjusts the RF environment to ensure the best radio connection and QoS, while mitigating co-channel and adjacent channel interference. Starting with AubaOS 8.0 in the Mobility Master — the next generation of master controller — static channel assignments in high density environments are now a thing of the past. Via machine learning, optimal channel, channel-width and power assignments are performed automatically to maximize capacity across the entire Wi-Fi network.

The channel or power assignment feature automatically assigns channel and power settings for all the Instant APs in the network according to changes in the RF environment. This feature automates many setup tasks during network installation and the ongoing operations when RF conditions change.

- **Min transmit power** -Specify the minimum transmission power. The value specified for **Min transmit power** indicates the minimum EIRP that can range from 3 dBm to 33 dBm in 3 dBm increments. If the minimum transmission EIRP setting configured on aninstant AP is not supported by the Instant AP model, this value is reduced to the highest supported power setting. The default value for minimum transmit power is 18 dBm.
- **Max transmit power** - Specify the maximum transmission power. The value specified for **Max transmit power** indicates the maximum EIRP that can range from 3 dBm to 33 dBm in 3 dBm increments. If the maximum transmission EIRP configured on aninstant AP is not supported by the Instant AP model, the value is reduced to the highest supported power setting. The default value for maximum transmit power is 127 dBm.

To determine interference, the ARM algorithm uses multiple pieces of information that the APs and AMs collect. These devices scan all available channels in the domain. The APs and AMs collect information on other APs, clients, rogue APs, background noise, and non-802.11 interference. These calculations feed into two indexes used by ARM: the interference index and coverage indices (see Figure 18).

Figure 18 Coverage and Interference Index

The interference index is used to monitor channel activity and interference. When the interference index is high compared to other channels, the AP looks to switch to a channel with a lower interference index. The coverage index is used to determine power levels for the AP. APs monitor other APs on the same channel, and ARM sets the AP power level based on the received transmission strength of other APs.

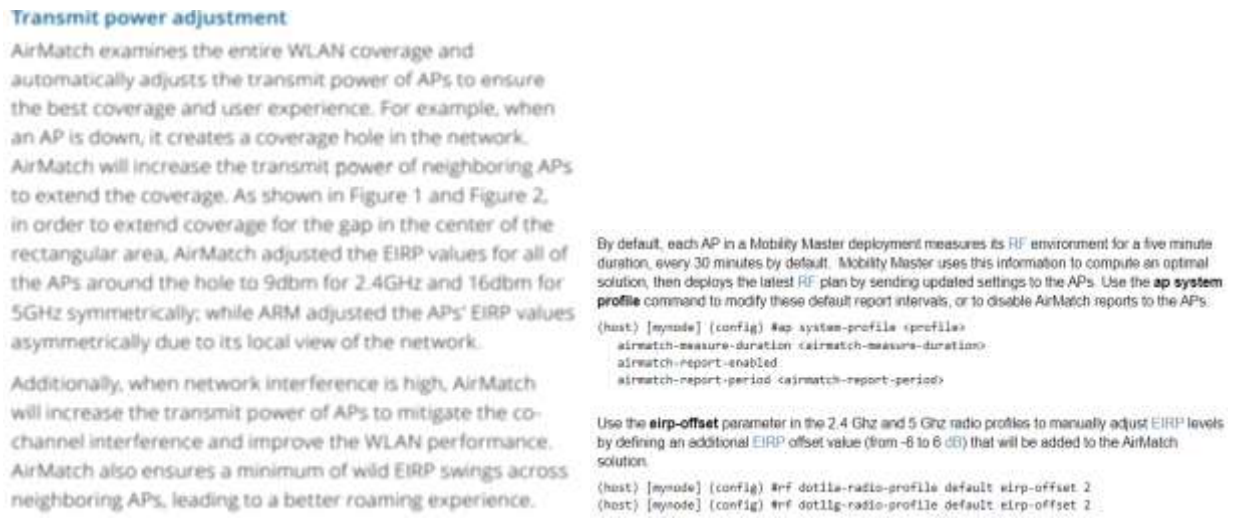
ARM monitors these two indices continually and is able to adapt automatically to a changing RF environment. Channel and power selection are adjusted automatically, without network administrator intervention. Except in the case of extreme interference or radar detection in certain channels, APs can be set to remain on a channel serving clients, thereby avoiding the disruption of a channel change.

Two metrics are maintained for every channel on every AP: the "coverage index" and the "interference index". These indices are used to calculate the optimum channel as well as transmit power for the AP.

Figure EC4-1 ARM Channel and Transmit Power Selection Algorithm

If APs detect strong signals from other APs on the same channel, they may decrease their power levels accordingly. Issue the CLI commands **show ap arm rf-summary ap-name <ap-name>** or **show ap arm rf-summary ip-addr <ap ip address>** for all APs and check their current coverage index (cov-idx). If the AP's coverage index is

The controller then computes the most optimized setting and instructs the AP to work accordingly. There is a small amount of time when the AP's leave their channel to conduct scanning operations that are essential to ARM. Where there are real time applications such as video/voice running on the network, this periodic scan might result in latency, jitter and eventually degrade the quality of the signal.



Transmit power adjustment

AirMatch examines the entire WLAN coverage and automatically adjusts the transmit power of APs to ensure the best coverage and user experience. For example, when an AP is down, it creates a coverage hole in the network. AirMatch will increase the transmit power of neighboring APs to extend the coverage. As shown in Figure 1 and Figure 2, in order to extend coverage for the gap in the center of the rectangular area, AirMatch adjusted the EIRP values for all of the APs around the hole to 9dbm for 2.4GHz and 16dbm for 5GHz symmetrically; while ARM adjusted the APs' EIRP values asymmetrically due to its local view of the network.

Additionally, when network interference is high, AirMatch will increase the transmit power of APs to mitigate the co-channel interference and improve the WLAN performance. AirMatch also ensures a minimum of wild EIRP swings across neighboring APs, leading to a better roaming experience.

By default, each AP in a Mobility Master deployment measures its RF environment for a five minute duration, every 30 minutes by default. Mobility Master uses this information to compute an optimal solution, then deploys the latest RF plan by sending updated settings to the APs. Use the **ap system profile** command to modify these default report intervals, or to disable AirMatch reports to the APs.

```
(host) [mynode] (config) #ap system-profile <profile>
  airmatch-measure-duration <airmatch-measure-duration>
  airmatch-report-enabled
  airmatch-report-period <airmatch-report-period>
```

Use the **eirp-offset** parameter in the 2.4 Ghz and 5 Ghz radio profiles to manually adjust EIRP levels by defining an additional EIRP offset value (from -8 to 6 dB) that will be added to the AirMatch solution.

```
(host) [mynode] (config) #rf dot11a-radio-profile default eirp-offset 2
(host) [mynode] (config) #rf dot11g-radio-profile default eirp-offset 2
```

AirMatch Benefits	
Even Channel assignment	Provides even distribution of radios across available channels, interference mitigation and maximized system capacity.
Dynamic Channel width adjustment	Dynamically adjusts between 20MHz, 40MHz and 80MHz to match the density of your environment.
Automatic transmit power adjustment	Examines the entire WLAN coverage and automatically adjusts the transmit power of APs to ensure the best coverage and user experience.

166. Additionally, HPE has been, and currently is an active inducer of infringement of the ’991 patent under 35 U.S.C. § 271(b) and contributory infringement of the ’991 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

167. HPE has actively induced, and continues to actively induce, infringement of the ’991 patent by intending that others use, offer for sale, sell and/or import products and/or services covered by the claim of the ’991 patent, including but not limited to the Accused ’991 Products and any other HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States products and/or services that directly infringe the claim of the ’991 patent as described above.

168. HPE has contributed to, and continues to contribute to, the infringement of the ’991 patent by others by offering to sell, selling, or otherwise commercially offering products and/or

services that, when installed and configured result in a system as intended by HPE, that directly infringes the claim of the '991 patent.

169. HPE knew of the '991 patent, or should have known of the '991 patent, but was willfully blind to its existence. HPE has had actual knowledge of the '991 patent since at least as early as service upon HPE of this Complaint. Additionally, HPE was aware of the '991 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the '991 patent.

170. HPE has committed, and continues to commit, affirmative acts that cause infringement of the claim of the '991 patent with knowledge of the '991 patent and knowledge or willful blindness that the induced acts constitute infringement of the claim of the '991 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customer way as desired and intended by HPE, infringes the claim of the '991 patent and/or by directly or indirectly providing instruction on how to use its products and/or services in a manner or configuration that infringes the claim of the '991 patent, including those found at the following:

- https://www.arubanetworks.com/assets/tg/TD_ArubaOS-8-Fundamental-Guide.pdf
- https://www.arubanetworks.com/techdocs/ArubaOS_80_Web_Help/Content/ArubaFrameStyles/ARM/mCell.htm
- https://cdw-prod.adobecqms.net/content/dam/cdw/on-domain-cdw/brands/aruba/aruba-os-8.pdf
- https://tdhpe.techdata.eu/Documents/Aruba/TB_AirMatch.pdf?epslanguage=it
- https://www.arubanetworks.com/assets/tg/TB_AirMatch.pdf
- https://www.arubanetworks.com/techdocs/ArubaOS_80_Web_Help/Content/ArubaFrameStyles/ARM/ConfiguringAirMatch.htm

- https://support.hpe.com/hpesc/public/docDisplay?docId=emr_na-a00018692en_us
- https://cc.cnetcontent.com/vcs/hp-ent/inline-content/VL/F/6/F6247B067D75150B198CAD97874FF5A02444C6B6_source.PDF
- https://support.hpe.com/hpesc/public/docDisplay?docId=a00044477en_us&docLocale=en_US
- https://www.slideshare.net/ArubaNetworks/aruba-80211n-networks-validated-reference-design
- https://www.arubanetworks.com/techdocs/ArubaOS_86_Web_Help/Content/arubaos-solutions/arm/arm-covr-inter-metr.htm?Highlight=arm%20metrices

171. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that, when used, cause the direct infringement of the claim of the '991 Patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such as software, that are especially made or especially adapted for use in infringement of the '991 patent and are not staple articles or commodities of commerce suitable for substantial non-infringing use.

172. As a result of HPE's acts of infringement, IV has suffered and will continue to suffer damages in an amount to be paid at trial.

COUNT VII
(HPE's Infringement of U.S. Patent No. 8,725,132)

173. Paragraphs 1-172 are incorporated by reference as if fully set forth herein.

174. The inventions claimed by the '132 patent, taken alone or in combination, were not well-understood, routine, or conventional to one of ordinary skill in the art at the time of the invention of the '132 patent. Rather, the patent teaches and claims an improved way to optimize wireless network performance particularly in crowded and dense wireless environments with lots

of user devices (stations) and network access points (APs) competing for limited resources. The inventions improved upon then-existing techniques for the configuration and management of wireless networks, which often required manual site engineering to determine and control the placement of APs in a given area and lacked the ability to efficiently and effectively share or reuse the same frequency channel with other nearby APs and stations, resulting in interference and sub-optimal resource usage. The inventions improved prior art wireless communications environments and systems by using an approach that enabled an AP to perform repeated power adjustment of the transmit power of that AP, as well as to communicate to an associated device (*e.g.*, an end user station, such as a smartphone or laptop associated with that AP) information regarding the selected transmit power at which that device's variable power transmitter should be set in order to reduce interference. This approach is particularly useful when a large number of APs and stations are operating in close proximity to one another, as it results in the repeated adjustment of transmit power of APs and stations so as to reduce interference in their network, thus allowing more APs and stations to operate in close proximity to each other.

175. The inventions represented a technical solution to an unsolved technological problem. The specification of the '132 patent describes, in technical detail, each of the limitations in the claim, allowing a person of skill in the art to understand what those limitations cover, and therefore what was claimed, and also understand how the non-conventional and non-generic ordered combination of the elements of the claim differs markedly from what had been conventional or generic in the industry at the time of the invention of the '132 patent. More specifically, the claims of the '132 patent require selecting an optimal power level up to a maximum power level at which a first device (*e.g.*, an AP) is to transmit signals, repeatedly adjusting the selected power level in order to reduce interference and transmitting a signal with

information indicative of a power level at which an associated device's variable power transmitter should be set in order to reduce interference.

176. The systems and methods covered by the claims of the '132 patent therefore differ markedly from the conventional and generic systems in use at the time of this invention, which lacked the above noted combination of selecting a transmit power level for a first device and repeatedly adjusting that power level to reduce interference, causing the first device to transmit at the selected power level, and causing the first device to send to a second device information indicative of a power level at which that associated device's variable power transmitter should be set in order to reduce interference.

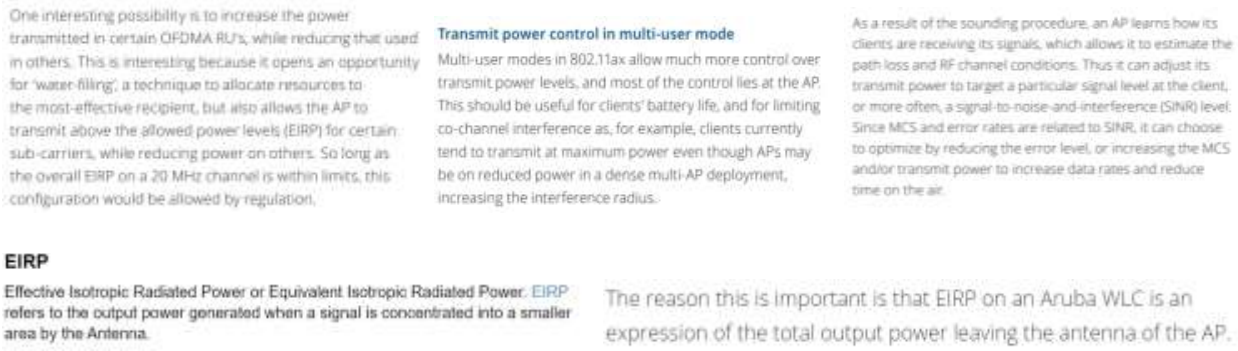
177. As described above, the '132 patent is drawn to solving a specific, technical problem arising in the context of wireless network management and configuration. Consistent with the problem addressed being rooted in such wireless network communications technology, the solutions provided by the '132 patent naturally are also rooted in that same technology and cannot be performed with pen and paper or in the human mind.

178. HPE has directly infringed, and continues to directly infringe, literally and/or by the doctrine of equivalents, individually and/or jointly, at least claim 2 of the '132 patent by making, using, testing, selling, offering for sale and/or importing products and/or services covered by the '132 patent. HPE's products and/or services that infringe the '132 patent include, but are not limited to, the Mobility Master feature, particularly the AirMatch Workflow functionality, of ArubaOS 8 employing Wi-Fi 6 and Wi-Fi 6E technology (and all versions/editions that support such functionality), including Aruba's 802.11ax, 530 Series, and 550 Series Access Points (APs), and any other of HPE's products and/or services, either alone or in combination, that operate in substantially the same manner (together the "Accused '132 Products").

Claim 1 of the ’132 patent is reproduced below:

> *1. Apparatus, comprising:*
> *control circuitry; and*
> *a variable power transmitter operable in response to the control circuitry to transmit wireless communication signals at a selected power level up to a maximum power level,*
> *wherein the selected power level is repeatedly adjusted in order to reduce interference, and to transmit a signal with information indicative of a power level at which a variable power transmitter of an associated device should be set in order to reduce interference.*

179. The Accused ’132 Products comprise an apparatus comprising control circuitry, and a variable power transmitter operable in response to the control circuitry to transmit wireless communication signals at a selected power level up to a maximum power level. For example, the Accused ’132 Products include APs that support the latest in Wi-Fi 6 standards and artificial intelligence (AI) capabilities. The Accused ’132 Products have substantial control over transmit power levels, through intelligent control of transmit power levels. The Accused ’132 Products learn how stations are receiving signals from a given AP, allowing the AP to estimate the path loss and RF channel conditions, and ultimately enabling it to adjust its transmit power to target a particular signal level at the station, typically a signal-to-interference-plus-noise (SINR) level:



180. The Accused ’132 Products comprise an apparatus wherein the selected power level is repeatedly adjusted in order to reduce interference. For example, in the Accused ’132 Products, multi-user modes in 802.11ax allow more control over transmit power level, employing techniques

such as a sounding procedure that allows an AP to determine how its user stations are receiving its signal, to estimate path loss and RF channel conditions, and to repeatedly adjust its transmit power to a target signal level or SINR at the user station:

Transmit power control in multi-user mode

Multi-user modes in 802.11ax allow much more control over transmit power levels, and most of the control lies at the AP. This should be useful for clients' battery life, and for limiting co-channel interference as, for example, clients currently tend to transmit at maximum power even though APs may be on reduced power in a dense multi-AP deployment, increasing the interference radius.

As a result of the sounding procedure, an AP learns how its clients are receiving its signals, which allows it to estimate the path loss and RF channel conditions. Thus it can adjust its transmit power to target a particular signal level at the client, or more often, a signal-to-noise-and-interference (SINR) level. Since MCS and error rates are related to SINR, it can choose to optimize by reducing the error level, or increasing the MCS and/or transmit power to increase data rates and reduce time on the air.

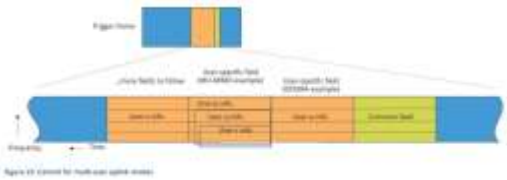
181. The Accused '132 Products comprise an apparatus that transmits a signal with information indicative of a power level at which a variable power transmitter of an associated device should be set in order to reduce interference. For example, the Accused '132 Products, which implement Wi-Fi 6 functionality, utilize trigger frames that contain information about the required received signal strength at the AP for each user station's transmission. Trigger frames are thus used to provide information indicative of a power level at which a variable power transmitter of an associated device should be set. Wi-Fi 6 user stations will increase or decrease their transmit power levels within a certain response time according to downlink signals from an AP. The Accused '132 Products send signals with information used to control associated user stations' transmit characteristics, including their transmit power levels.

The trigger frame format is shown above. It contains the following information.

- The length of the uplink transmission window
- Which client devices are to transmit
- Which OFDMA RU's are to be used by each client device
- How many spatial streams are to be used by each client device
- Which MCS modulation level is to be used by each client device
- Whether functions like STBC are to be used on the uplink
- Required signal strength at the AP for the client's transmission (this is calculated by the client using the AP's transmit power level, the client's receive RSSI level and an assumption of channel reciprocity)
- (Uplink MU-MIMO is deferred to wave 2, but all the necessary fields are already defined in the 802.11ax standard.)

subcarriers within the same channel. Uplink OFDMA is more difficult to manage than the downlink variety, as many different clients must be coordinated: the access point transmits trigger frames to indicate which sub-channels each client can use.

The trigger frame is used to map clients to their respective OFDMA RU's and MU-MIMO groups, and includes timing and MCS modulation-rate information, as well as transmit-power guidance.



sensitivity limits of the AP. If the power levels of multiple users are not balanced, network performance will be compromised by intercarrier interference (ICI) and compression when a Wi-Fi receiver attempts to process multiple signals across a wide dynamic range. Wi-Fi 6 devices will increase or decrease their transmit power levels within a certain response time according to downlink signals from an AP.

With the new multi-user signaling mechanisms, the AP can now control the client's transmit characteristics. In wave 1 this is only applicable to OFDMA, but in wave 2, uplink MU-MIMO will be controlled as well. The fields controlling uplink multi-user operation allow the AP to specify the transmit power indirectly, as a desired SINR at the AP, which the client can derive from the sounding estimate of path loss. But more than that, the AP specifies the number of spatial streams to use, the MCS, the OFDMA RU and other features that should be used. The AP may do this to optimize packing of an OFDMA packet, but alternatively it could seek to reduce the transmit power used by particular clients, for battery life or interference optimization purposes.

182. Additionally, HPE has been, and currently is an active inducer of infringement of the ’132 patent under 35 U.S.C. § 271(b) and contributory infringement of the ’132 patent under 35 U.S.C. § 271(c) either literally and/or by the doctrine of equivalents.

183. HPE has actively induced, and continues to actively induce, infringement of the ’132 patent by intending that others use, offer for sale, sell and/or import products and/or services covered by the ’132 patent, including but not limited to the Accused ’132 Products and any other HPE product and/or service, alone or in combination, that operates in materially the same manner. HPE provides these products and/or services to others, such as customers, resellers and end-user customers, who, in turn, in accordance with HPE’s design, intent and directions, use, provision for use, offer for sale, or sell in the United States products and/or services that directly infringe the ’132 patent as described above.

184. HPE has contributed to, and continues to contribute to, the infringement of the ’132 patent by others by offering to sell, selling, or otherwise commercially offering products and/or services that, when installed and configured result in a system as intended by HPE, that directly infringes the ’132 patent.

185. HPE knew of the ’132 patent, or should have known of the ’132 patent, but was willfully blind to its existence. HPE has had actual knowledge of the ’132 patent since at least as early as service upon HPE of this Complaint. Additionally, HPE was aware of the ’132 patent at least by March 8, 2021, as a result of correspondence directed to HPE by IV. By the time of trial, HPE will have known and intended (since receiving such notice) that its continued actions would infringe and actively induce and contribute to the infringement of the ’132 patent.

186. HPE has committed, and continues to commit, affirmative acts that cause infringement of the ’132 patent with knowledge of the ’132 patent and knowledge or willful blindness that the induced acts constitute infringement of the ’132 patent. As an illustrative example only, HPE induces such acts of infringement by its affirmative actions of intentionally providing hardware and/or software components that when used in their normal and customer way as desired and intended by HPE, infringes the ’132 patent and/or by directly or indirectly providing instruction on how to use its products and/or services in a manner or configuration that infringes the ’132 patent, including those found at the following:

- https://www.arubanetworks.com/assets/wp/WP_802.11AX.pdf
- https://www.arubanetworks.com/assets/so/SO_80211ax.pdf

187. HPE has also committed, and continues to commit, contributory infringement by knowingly offering to sell, selling, or otherwise commercializing products and/or services that, when used, cause the direct infringement of the ’132 patent by a third party, and which have no substantial non-infringing uses, or include one or more separate and distinct components, such as software, that are especially made or especially adapted for use in infringement of the ’132 patent and are not staple articles or commodities of commerce suitable for substantial non-infringing use.

188. As a result of HPE's acts of infringement, IV has suffered and will continue to suffer damages in an amount to be paid at trial.

PRAYER FOR RELIEF

IV requests that the Court enter judgment against HPE as follows:

(A) finding that HPE has infringed one or more claims of each of the asserted patents, directly and/or indirectly, literally and/or under the doctrine of equivalents;

(B) awarding damages sufficient to compensate IV for HPE's infringement under 35 U.S.C. § 284;

(C) finding this case exceptional under 35 U.S.C. § 285 and awarding IV its reasonable attorneys' fees;

(D) awarding IV its costs and expenses incurred in this action;

(E) awarding IV prejudgment and post-judgment interest; and

(F) granting IV such further relief as the Court deems just and appropriate.

DEMAND FOR JURY TRIAL

IV demands trial by jury of all claims so triable under Federal Rule of Civil Procedure 38.

Date: March 9, 2021

Respectfully submitted,

/s/ Derek Gilliland
DEREK GILLILAND
STATE BAR NO. 24007239
SOREY, GILLILAND & HULL, LLP
P.O. Box 4203
109 W. Tyler St.
Longview, Texas 75601
903.212.2822 (telephone)
derek@soreylaw.com

OF COUNSEL:

Paul J. Hayes
phayes@princelobel.com
Matthew Vella
mvella@princelobel.com
Robert R. Gilman
rgilman@princelobel.com
Jonathan DeBlois
jdeblois@princelobel.com
Thomas Fulford
tfulford@princelobel.com
PRINCE LOBEL TYE LLP
One International Place, Suite 3700
Boston, MA 02110
Tel: (617) 456-8000
Fax: (617) 456-8100

COUNSEL for PLAINTIFFS