US 20200050472A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2020/0050472 A1

KEDEM (43) Pub. Date: Feb. 13, 2020

(54) METHODS AND APPARATUS FOR PROVIDING HYPERVISOR LEVEL DATA SERVICES FOR SERVER VIRTUALIZATION

(71) Applicant: **Zerto Ltd.**, Herzilya (IL)

(72) Inventor: **Ziv KEDEM**, Herzilya (IL)

(21) Appl. No.: **16/654,557**

(22) Filed: **Oct. 16, 2019**

Related U.S. Application Data

(63) Continuation of application No. 15/240,847, filed on Aug. 18, 2016, now Pat. No. 10,459,749, which is a continuation of application No. 13/039,446, filed on Mar. 3, 2011.

(60) Provisional application No. 61/314,589, filed on Mar. 17, 2010.

Publication Classification

(51) **Int. Cl.**

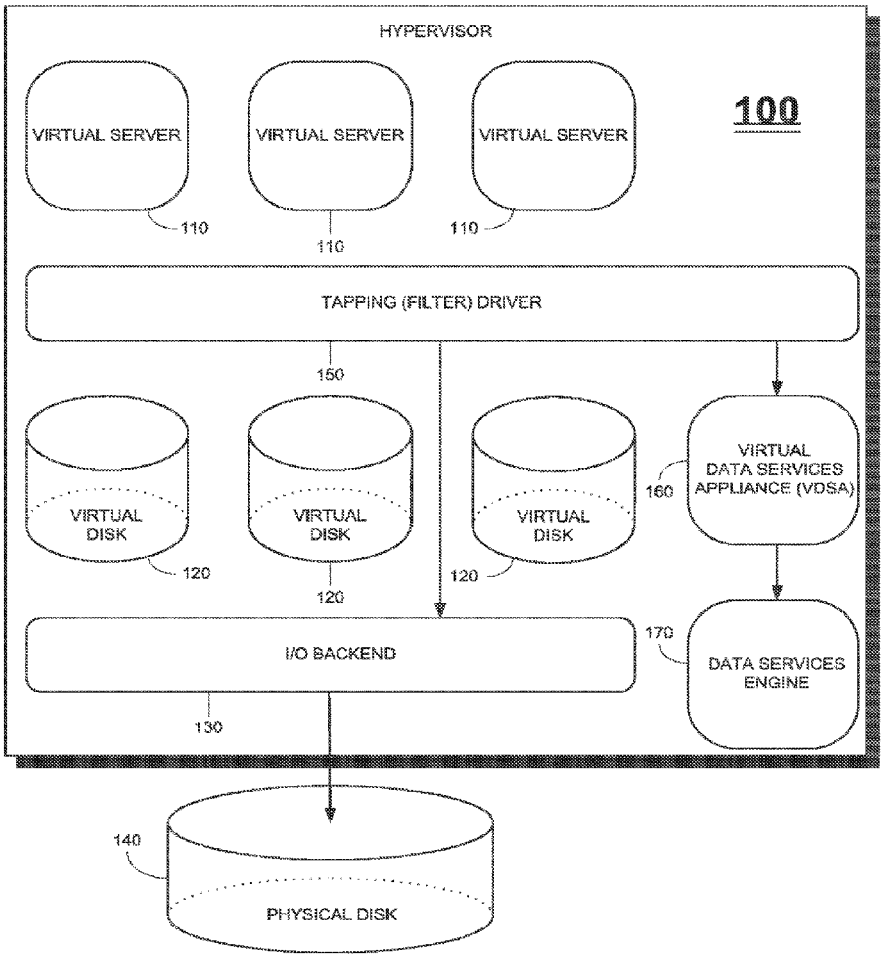
G06F 9/455	(2006.01)
G06F 3/06	(2006.01)
G06F 9/54	(2006.01)

(52) **U.S. Cl.**

CPC ***G06F 9/45558*** (2013.01); ***G06F 3/067*** (2013.01); ***G06F 3/0665*** (2013.01); ***G06F 3/0653*** (2013.01); ***G06F 9/545*** (2013.01); *G06F 11/1471* (2013.01); ***G06F 3/0619*** (2013.01); ***G06F 3/0689*** (2013.01); *G06F 2009/45579* (2013.01); *G06F 2209/542* (2013.01); ***G06F 9/45533*** (2013.01)

(57) **ABSTRACT**

A hypervisor virtual server system, including a plurality of virtual servers, a plurality of virtual disks that are read from and written to by the plurality of virtual servers, a physical disk, an I/O backend coupled with the physical disk and in communication with the plurality of virtual disks, which reads from and writes to the physical disk, a tapping driver in communication with the plurality of virtual servers, which intercepts I/O requests made by any one of said plurality of virtual servers to any one of said plurality of virtual disks, and a virtual data services appliance, in communication with the tapping driver, which receives the intercepted I/O write requests from the tapping driver, and that provides data services based thereon.



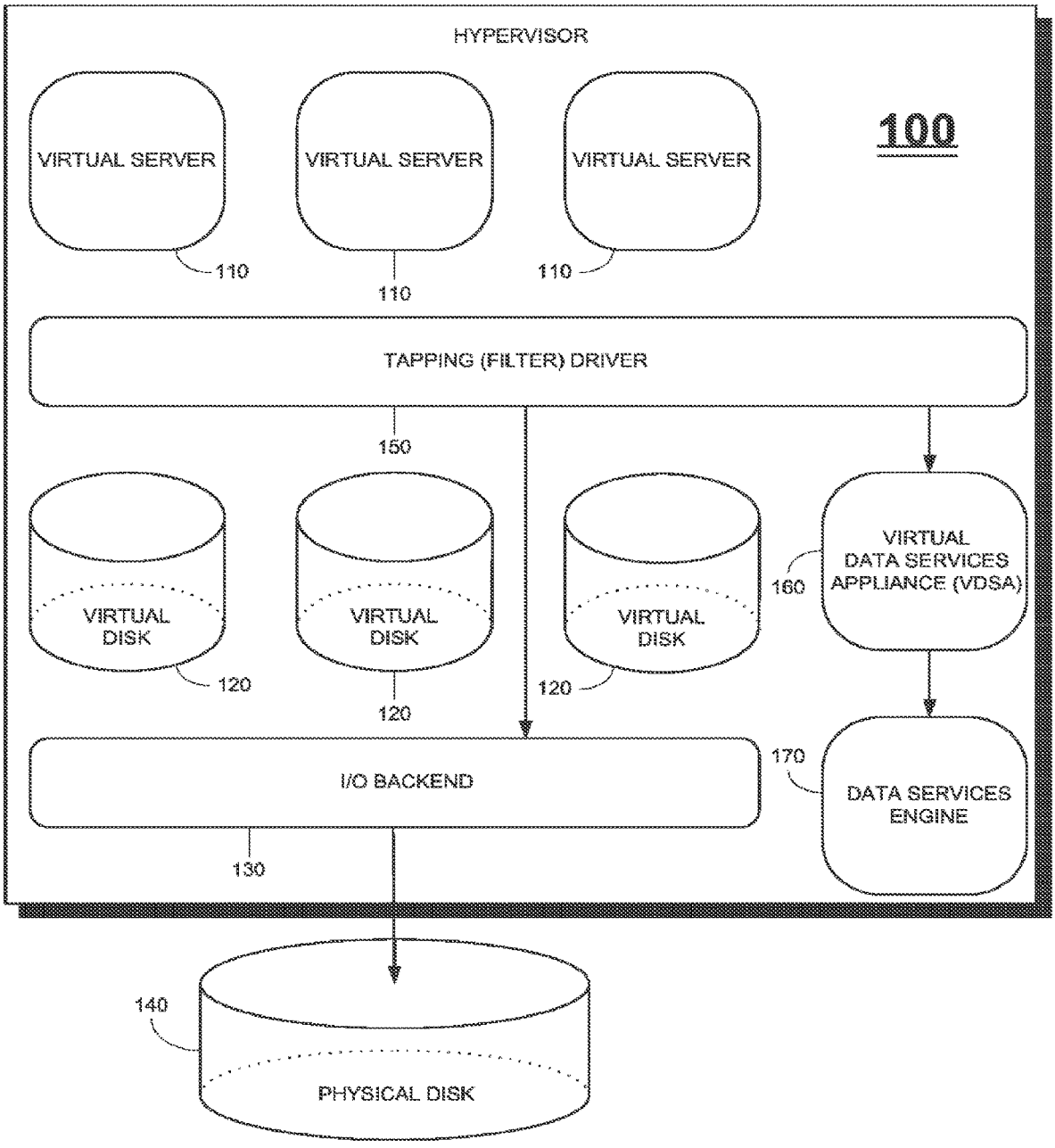

HYPERVISOR
100
VIRTUAL SERVER
VIRTUAL SERVER
VIRTUAL SERVER
110
110
110
TAPPING (FILTER) DRIVER
150
VIRTUAL DISK
VIRTUAL DISK
VIRTUAL DISK
120
120
120
160
VIRTUAL DATA SERVICES APPLIANCE (VDSA)
170
DATA SERVICES ENGINE
I/O BACKEND
130
140
PHYSICAL DISK

FIG. 1

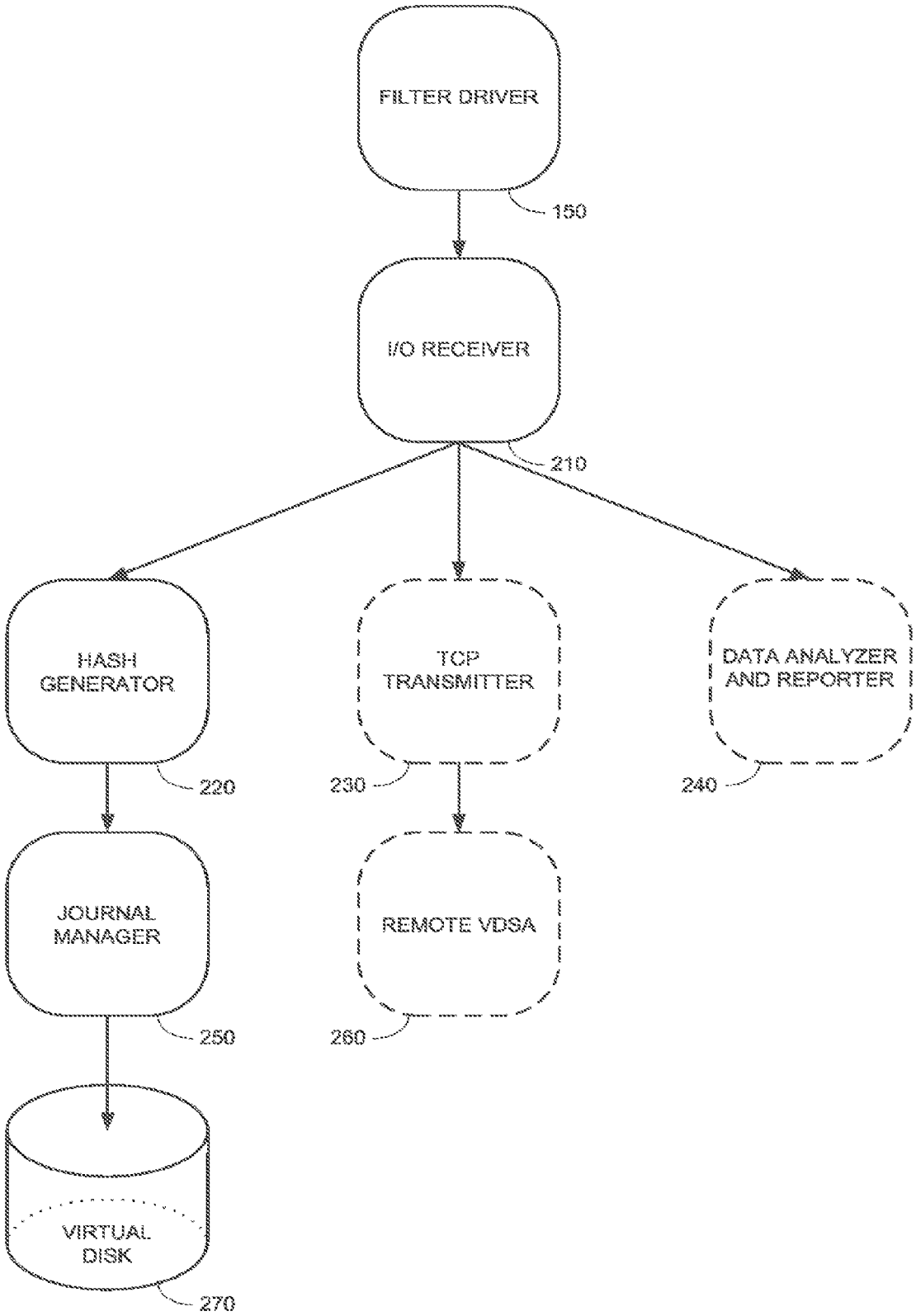

FILTER DRIVER
150
I/O RECEIVER
210
HASH GENERATOR
220
TCP TRANSMITTER
230
DATA ANALYZER AND REPORTER
240
JOURNAL MANAGER
250
REMOTE VDSA
260
VIRTUAL DISK
270

FIG. 2

METHODS AND APPARATUS FOR PROVIDING HYPERVISOR LEVEL DATA SERVICES FOR SERVER VIRTUALIZATION

CROSS REFERENCES TO RELATED APPLICATIONS

[0001] This application is a continuation of, and claims priority under 35 U.S.C. § 120 and benefit of U.S. patent application Ser. No. 15/240,847, titled METHODS AND APPARATUS FOR PROVIDING HYPERVISOR LEVEL DATA SERVICES FOR SERVER VIRTUALIZATION, filed Aug. 18, 2016 by inventor Ziv Kedem, which is a continuation of, and claims priority under 35 U.S.C. § 120 and benefit of U.S. patent application Ser. No. 13/039,446, titled METHODS AND APPARATUS FOR PROVIDING HYPERVISOR LEVEL DATA SERVICES FOR SERVER VIRTUALIZATION, filed Mar. 3, 2011 by inventor Ziv Kedem, which claims priority under 35 U.S.C. § 119 and benefit of U.S. Provisional Application No. 61/314,589, titled METHODS AND APPARATUS FOR PROVIDING HYPERVISOR LEVEL DATA SERVICES FOR SERVER VIRTUALIZATION, filed on Mar. 17, 2010 by inventor Ziv Kedem, each of which is incorporated by reference in its entirety.

FIELD OF THE INVENTION

[0002] The present invention relates to virtual server environments and data services.

BACKGROUND OF THE INVENTION

[0003] Virtual servers are logical entities that run as software in a server virtualization infrastructure, referred to as a "hypervisor". Examples of hypervisors are VMWARE® ESX manufactured by VMware, Inc. of Palo Alto, Calif., HyperV manufactured by Microsoft Corporation of Redmond, Wash., XENSERVER manufactured by Citrix Systems, Inc. of Fort Lauderdale, Fla., Redhat KVM manufactured by Redhat, Inc. of Raleigh, N.C., and Oracle VM manufactured by Oracle Corporation of Redwood Shores, Calif. A hypervisor provides storage device emulation, referred to as "virtual disks", to virtual servers. Hypervisor implements virtual disks using back-end technologies such as files on a dedicated file system, or raw mapping to physical devices.

[0004] As distinct from physical servers that run on hardware, virtual servers run their operating systems within an emulation layer that is provided by a hypervisor. Although virtual servers are software, nevertheless they perform the same tasks as physical servers, including running server applications such as database applications, customer relation management applications and MICROSOFT EXCHANGE SERVER®. Most applications that run on physical servers are portable to run on virtual servers. As distinct from virtual desktops that run client side applications and service individual users, virtual servers run applications that service a large number of clients.

[0005] As such, virtual servers depend critically on data services for their availability, security, mobility and compliance requirements, the data services including inter alia continuous data protection, disaster recovery, remote replication, data security, mobility, and data retention and archiving policies

SUMMARY OF THE DESCRIPTION

[0006] Aspects of the present invention relate to a dedicated virtual data service appliance (VDSA) within a hypervisor that can provide a variety of data services. Data services provided by a VDSA include inter alia replication, monitoring and quality of service.

[0007] In an embodiment of the present invention, a tapping filter driver is installed within the hypervisor kernel. The tapping driver has visibility to I/O requests made by virtual servers running on the hypervisor.

[0008] A VDSA runs on each physical hypervisor. The VDSA is a dedicated virtual server that provides data services; however, the VDSA does not necessarily reside in the actual I/O data path. When a data service processes I/O asynchronously, the VDSA receives the data outside the data path.

[0009] Whenever a virtual server performs I/O to a virtual disk, the tapping driver identifies the I/O requests to the virtual disk. The tapping driver copies the I/O requests, forwards one copy to the hypervisor's backend, and forwards another copy to the VDSA.

[0010] Upon receiving an I/O request, the VDSA performs a set of actions to enable various data services. A first action is data analysis, to analyze the data content of the I/O request and to infer information regarding the virtual server's data state. E.g., the VDSA may infer the operating system level and the status of the virtual server. This information is subsequently used for reporting and policy purposes.

[0011] A second action, optionally performed by the VDSA, is to store each I/O write request in a dedicated virtual disk for journaling. Since all I/O write requests are journaled on this virtual disk, the virtual disk enables recovery data services for the virtual server, such as restoring the virtual server to an historical image.

[0012] A third action, optionally performed by the VDSA, is to send I/O write requests to different VDSAs, residing on hypervisors located at different locations, thus enabling disaster recovery data services.

[0013] There is thus provided in accordance with an embodiment of the present invention a hypervisor virtual server system, including a plurality of virtual servers, a plurality of virtual disks that are read from and written to by the plurality of virtual servers, a physical disk, an I/O backend coupled with the physical disk and in communication with the plurality of virtual disks, which reads from and writes to the physical disk, a tapping driver in communication with the plurality of virtual servers, which intercepts I/O requests made by any one of said plurality of virtual servers to any one of said plurality of virtual disks, and a virtual data services appliance, in communication with the tapping driver, which receives the intercepted I/O write requests from the tapping driver, and which provides data services based thereon.

[0014] There is additionally provided in accordance with an embodiment of the present invention a method for providing data services within a hypervisor virtual server system, including intercepting I/O requests from any one of a plurality of virtual servers to one of a plurality of virtual disks, and sending intercepted I/O write requests to a virtual data services appliance that provides hypervisor data services.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The present invention will be more fully understood and appreciated from the following detailed description, taken in conjunction with the drawings in which:

[0016] FIG. **1** is a simplified block diagram of a hypervisor architecture that includes a tapping driver and a virtual data services appliance, in accordance with an embodiment of the present invention; and

[0017] FIG. **2** is a simplified data flow chart for a virtual data services appliance, in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

[0018] Aspects of the present invention relate to a dedicated virtual data services appliance (VDSA) within a hypervisor, which is used to provide a variety of hypervisor data services. Data services provided by a VDSA include inter alia replication, monitoring and quality of service.

[0019] Reference is made to FIG. **1**, which is a simplified block diagram of a hypervisor architecture that includes a tapping driver and a VDSA, in accordance with an embodiment of the present invention. Shown in FIG. **1** is a hypervisor **100** with three virtual servers **110**, three virtual disks **120**, an I/O backend **130** and a physical storage array **140**. Hypervisor **100** uses a single physical server, but runs multiple virtual servers **110**. Virtual disks **120** are a storage emulation layer that provide storage for virtual servers **110**. Virtual disks **120** are implemented by hypervisor **100** via I/O backend **130**, which connects to physical disk **140**.

[0020] Hypervisor **100** also includes a tapping driver **150** installed within the hypervisor kernel. As shown in FIG. **1**, tapping driver **150** resides in a software layer between virtual servers **110** and virtual disks **120**. As such, tapping driver **150** is able to access I/O requests performed by virtual servers **110** on virtual disks **120**. Tapping driver **150** has visibility to I/O requests made by virtual servers **110**.

[0021] Hypervisor **100** also includes a VDSA **160**. In accordance with an embodiment of the present invention, a VDSA **160** runs on a separate virtual server within each physical hypervisor. VDSA **160** is a dedicated virtual server that provides data services via one or more data services engines **170**. However, VDSA **160** does not reside in the actual I/O data path between I/O backend **130** and physical disk **140**. Instead, VDSA **160** resides in a virtual I/O data path.

[0022] Whenever a virtual server **110** performs I/O on a virtual disk **120**, tapping driver **150** identifies the I/O requests that the virtual server makes. Tapping driver **150** copies the I/O requests, forwards one copy via the conventional path to I/O backend **130**, and forwards another copy to VDSA **160**. In turn, VDSA **160** enables the one or more data services engines **170** to provide data services based on these I/O requests.

[0023] Reference is made to FIG. **2**, which is a simplified data flow chart for a VDSA, in accordance with an embodiment of the present invention. Shown in FIG. **2** are an I/O receiver **210**, a hash generator **220**, a TCP transmitter **230**, a data analyzer and reporter **240**, a journal manager **250** and a remote VDSA **260**. Remote VDSA **260** resides on different physical hardware, at a possibly different location.

[0024] As shown in FIG. **2**, I/O receiver **210** receives an intercepted I/O request from tapping driver **150**. VDSA **160** makes up to three copies of the received I/O requests, in order to perform a set of actions which enable the one or more data services engines **170** to provide various services.

[0025] A first copy is stored in persistent storage, and used to provide continuous data protection. Specifically, VDSA **160** sends the first copy to journal manager **250**, for storage in a dedicated virtual disk **270**. Since all I/O requests are journaled on virtual disk **270**, journal manager **250** provides recovery data services for virtual servers **110**, such as restoring virtual servers **110** to an historical image. In order to conserve disk space, hash generator **220** derives a one-way hash from the I/O requests. Use of a hash ensures that only a single copy of any I/O request data is stored on disk.

[0026] An optional second copy is used for disaster recovery. It is sent via TCP transmitter **230** to remote VDSA **260**. As such, access to all data is ensured even when the production hardware is not available, thus enabling disaster recovery data services.

[0027] An optional third copy is sent to data analyzer and reporter **240**, which generates a report with information about the content of the data. Data analyzer and reporter **240** analyzes data content of the I/O requests and infers information regarding the data state of virtual servers **110**. E.g., data analyzer and reporter **240** may infer the operating system level and the status of a virtual server **110**.

[0028] In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made to the specific exemplary embodiments without departing from the broader spirit and scope of the invention as set forth in the appended claims. Accordingly, the specification and drawings are to be regarded in an illustrative rather than a restrictive sense.

What is claimed is:

1. A hypervisor, comprising:

a virtual server to make an I/O request via an I/**0** data path;

a virtual disk to be read and written to using the I/**0** request made by the virtual server via the I/O data path;

a tapping driver having visibility to the I/O data path, the tapping driver to intercept the I/O request made by the virtual server via the I/O data path to the virtual disk; and

a virtual data services appliance residing outside the I/O data path, in communication with the tapping driver to asynchronously receive the I/O request and provide data services based on the I/O request, and

the tapping driver to cause the I/O request to be forwarded along the I/O data path and to separately cause the I/O request to be forwarded to the virtual data services appliance.

2. The hypervisor of claim **1**, comprising a data services engine in communication with the virtual data services appliance to provide the data services based on the I/O request.

3. The hypervisor of claim **1**, comprising an I/O backend coupled with a physical disk in communication with the virtual disk to read to and write from the virtual disk.

4. The hypervisor of claim **1**, comprising a data analyzer in communication with the virtual data services appliance to determine a data state indicating a state of the virtual server based on content of the I/O request.

5. The hypervisor of claim **1**, comprising a journal manager in communication with the virtual data services appliance to store the I/O request as a journal entry in the virtual disk.

6. The hypervisor of claim **1**, comprising a hash generator in communication with the tapping driver to generate a hash value using the I/O request to store data corresponding to the I/O request onto the virtual disk.

7. The hypervisor of claim **1**, comprising an I/O receiver to receive the I/O request intercepted by the tapping driver, the I/O receiver to generate a plurality of copies of the I/O request to separately process each copy of the plurality of copies of the I/O request.

8. The hypervisor of claim **1**, comprising the data services engine to provide the data services based on a set of actions performed using a copy of the I/O request generated by the virtual data services appliance asynchronously to processing of the I/O request in the I/O data path.

9. The hypervisor of claim **1**, comprising the I/O data path separate from a virtual I/O data path, the I/O data path excluding the virtual data services appliance, the virtual I/O data path including the virtual data services appliance.

10. A system to provide hypervisor data services, comprising:

a virtual server implemented on a hypervisor to read from and write to a virtual disk via an input/output (I/O) data path;

a tapping driver implemented on the hypervisor in communication with the virtual server, the tapping driver having visibility to the I/O data path to intercept an I/O request from the virtual server to the virtual disk;

a virtual data services appliance implemented on the hypervisor residing outside the I/O data path, the virtual data services appliance in communication with the tapping driver to generate a copy of the I/O request intercepted by the tapping driver and to asynchronously perform a set of actions to provide data services based on the copy of the I/O request; and

the tapping driver to cause the I/O request to be forwarded along the I/O data path and to separately cause the I/O request to be forwarded to the virtual data services appliance.

11. The system of claim **10**, comprising a data services engine implemented on the hypervisor in communication with the virtual data services appliance to provide data services for the hypervisor based on the set of actions performed asynchronously using the copy of the I/O request.

12. The system of claim **10**, comprising a data analyzer implemented on the hypervisor in communication with the virtual data services appliance to determine a data state indicating a state of the virtual server based on content of the I/O request.

13. The system of claim **10**, comprising an I/O backend implemented on the hypervisor coupled with a physical disk in communication with the virtual disk to read to and write from the virtual disk.

14. The system of claim **10**, comprising an I/O receiver implemented on the hypervisor to receive the I/O request intercepted by the tapping driver, the I/O receiver to send a copy of the I/O request to a remote virtual data services appliance.

15. The system of claim **10**, comprising:

the tapping driver to forward a copy of the I/O request to the virtual data services appliance; and

a data services engine implemented on the hypervisor in communication with the virtual data services appliance to provide data services based on the copy of the I/O request received by the virtual data services appliance.

16. A method of providing hypervisor data services, comprising:

identifying, by a tapping driver implemented on a hypervisor, an input/output (I/O) request from a virtual server to a virtual disk via an I/O data path, the tapping driver having access to the I/O data path;

sending, by the tapping driver, the I/O request to a virtual data services appliance residing outside the I/O data path, the virtual data services appliance to provide data services via a data services engine based on the I/O request asynchronously to processing of the I/O request in the I/O data path; and

causing, by the tapping driver, the I/O request to be forwarded along the I/O data path separately from forwarding the I/O request to the virtual data services appliance.

17. The method of claim **16**, comprising sending, by the tapping driver, the I/O request to a journal manager, receipt of the I/O request causing the journal manager to store the I/O request as a journal entry in the virtual disk.

18. The method of claim **16**, comprising sending, by the tapping driver, the I/O request to the virtual data services appliance, receipt of the I/O request causing the virtual data services appliance to generate a copy of the I/O request and to perform at least one action of a set of actions based on the copy of the I/O request to provide the data services via the data services engine.

19. The method of claim **16**, comprising:

copying, by the tapping driver, the I/O request from the virtual server to the virtual disk via the I/O data path to generate a copy of the I/O request; and

forwarding, by the tapping driver, the I/O request the copy of the I/O request to the virtual data services appliance, receipt of the copy of the I/O request to cause the virtual data services appliance to provide the data services via the data services engine.

20. The method of claim **16**, comprising:

intercepting, by the tapping driver, the I/O request made by the virtual server to the virtual disk from the I/O data path; and

sending, by the tapping driver, the I/O request via a virtual I/O data path to the data services engine, the virtual I/O data path separate from the I/O data path, the virtual I/O data path including the virtual data services appliance.

* * * * *