



(51) International Patent Classification:
G06F 11/14 (2006.01)

MARINELLI, Claudio [IT/IT]; IBM Italia S.P.A., Via
Sciengai 53, I-00144 Roma (IT).

(21) International Application Number:
PCT/EP2010/055833

(74) Agent: **SCHIFANO, Nicolas**; IBM France, Direction de
la Propriété Intellectuelle, Route de Saint-Jeannet,
F-06610 La Gaude (FR).

(22) International Filing Date:
29 April 2010 (29.04.2010)

(81) Designated States (*unless otherwise indicated, for every
kind of national protection available*): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,
HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP,
KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD,
ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI,
NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD,
SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR,
TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09165126.5 10 July 2009 (10.07.2009) EP

(71) Applicant (*for all designated States except US*): **INTER-
NATIONAL BUSINESS MACHINES CORPORA-
TION** [US/US]; New Orchard Road, Armonk, New York
10504 (US).

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): **FONTIGNIE,
Jacques** [BE/CH]; IBM Switzerland Ltd, Rue De Bains
33-35, CH-1205 Geneve (CH). **VUILLEUMIER
STUECKELBERG, Marc** [CH/CH]; IBM Switzerland
Ltd, Rue De Bains 33-35, CH-1205 Geneve (CH).

(84) Designated States (*unless otherwise indicated, for every
kind of regional protection available*): ARIPO (BW, GH,
GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG,
ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ,
TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE,
ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV,
MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM,

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR CAPTURING AN IMAGE OF A SOFTWARE ENVIRONMENT

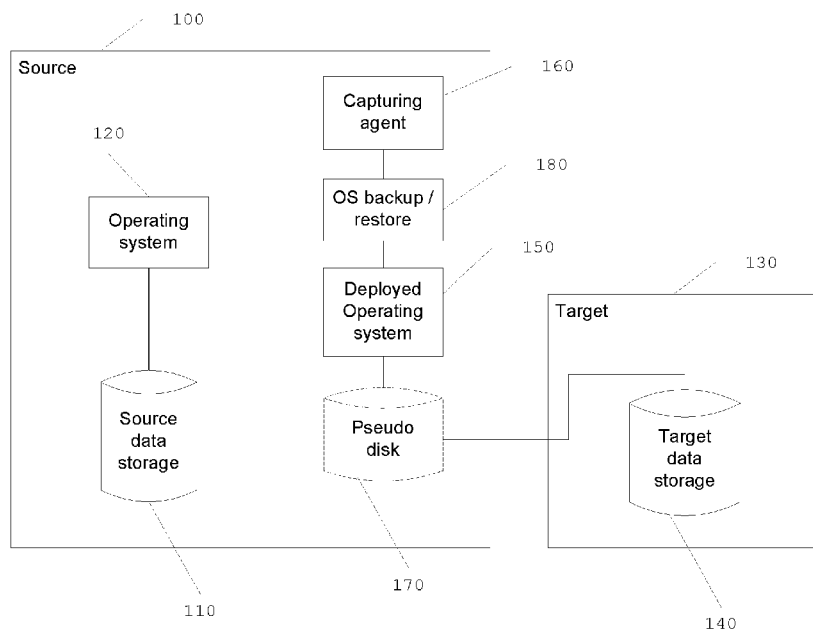


Figure 1

(57) Abstract: The invention provides a method and system for optimally capturing an image on a physical or virtual environment and transferring it to a target system, by leveraging the capabilities of the operating system residing on the source system to manipulate the filesystem of the source storage device, and by presenting the target data storage as a local simulated disk to the source environment.



TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, **Published:**
ML, MR, NE, SN, TD, TG).

— *with international search report (Art. 21(3))*

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

5

SYSTEM AND METHOD FOR CAPTURING AN IMAGE OF A SOFTWARE ENVIRONMENT

10

Field of the invention

The present invention relates to a method and system for capturing a software image from a source environment and transferring it to a target environment.

15 Background of the invention

As the complexity of IT systems and the time to install and configure an environment increases, there is a growing need to efficiently replicate images or environments across multiple systems.

At the most basic level, an image encapsulates the data, applications and operating systems that
20 reside on a physical or virtual machine. Organizations are beginning to view the data centre as a set of portable images instead of a diverse mix of a servers, operating systems, applications and data. The ability to profile, move, copy, protect and replicate these images between physical and virtual hosts is a key enabler for operational efficiency and business success.

One of the key tasks required to migrate an image is the ability to capture the image, moving it to
25 a central repository before customizing and deploying it to other targets. Different methods are available to capture an image each one with its advantages and its drawbacks.

It is possible to read the sectors of the source disk and copy them to the target disk. This is very inefficient as the source disk may be fragmented and some sectors can be empty.

An improved approach is to copy only the allocated sectors or blocks of the source disk.

However, if the source disk is fragmented, the process can be inefficient as useless data will be copied as well.

An alternative is to copy files and elements of the source system at the filesystem level. The

5 advantage is that the output is defragmented on the target system, and it is much faster to execute than the previous methods, as the amount of data to process is smaller. However the copy agents must be able to manage the filesystems natively. This is not always possible as there exist many different varieties of filesystems and they are often poorly documented.

10 These techniques can be applied to virtual machines as well. Virtual machines are often handled as files at the hypervisor level. This creates a lot of inefficiencies as the size of the virtual machine footprint on the disk can only grow as the hypervisor is in general unable to detect which hard disks blocks are not used by the virtual machine.

Summary of the invention

According to a first aspect of the present invention, there is provided a method for capturing a file stored on a first data storage device of a computer system, said file being encoded in a first filesystem format on said first data storage device, said computer system comprising a first operating system with a first application suitable for retrieving all the filesystem attributes associated with said file, said method comprising the steps of:

- receiving a first message indicating a locator of a second data storage device;
- creating on said computer system a handler to said second data storage device, so that said second data storage appears as a local device to said computer system;
- instructing said first application to retrieve the file data and all filesystem attributes associated with said file from said first data storage;
- instructing a second application to send said retrieved file data and said retrieved filesystem attributes to said handler so that said file data and said filesystem attributes are written on said second data storage.

An advantage of this aspect is that the target environment does not need to have any knowledge of the filesystem format in which the file that is included.

A further advantage of this aspect is that access to a network is not required as the target system on which the file would be replicated can be a local disk such as a USB disk.

Yet another advantage of this aspect is that, if the target system is a central repository for storing replicated images, then the server will not become a bottleneck when replicating several images at the same time as most of the processing would happen at the source sites before the data is actually written on to the device handler.

A further advantage of this aspect is that a software component without any knowledge of the underlying filesystem can leverage the operating system to replicate the filesystem attributes associated with the file on the target environment, this is particularly important when replicating an operating system which cannot operate without these attributes being set.

In a first development of the first aspect, the method comprises the further step of compressing the file data before sending it to said handler.

An advantage of the first development is that the file replication can happen faster.

In a second development of the first aspect, the method further comprises the steps of:

- receiving a message identifying the type of said first operating system;
 - deploying an agent for running a file capture on a second operating system which is of the same type as said first operating system so that it can manage said first filesystem format;
 - booting said second operating system on said computer system, so that said agent executes the
- 5 steps of the method of the first aspect.

An advantage of the second development is that any file can be captured and replicated without the risk of them being locked, and the agent can be agnostic with respect to the underlying filesystem which must be replicated.

In a third development of the first aspect, said second storage device is a remote disk.

10 An advantage is that environments can be duplicated easily over the network.

In a fourth development of the first aspect, the method further comprises the step of formatting said second storage location so that its filesystem format is the same as said first filesystem format.

15 An advantage is that the target environment does not require any knowledge of the filesystem of the source data storage. Furthermore, the embodiment enables formatting of a remote disk, which is not possible with a network drive.

In the fifth development of the first aspect, a partition is created on said second storage location and said partition is aligned on the block size of said first filesystem format.

Advantage is that the performance of the replicated environment is greatly increased.

20 In a sixth development of the first aspect, the steps of receiving and sending are respectively performed by a backup and restore application associated with the operating system.

An advantage is that these applications are usually provided by the operating system which has full knowledge of the filesystem, thus the applications have no difficulties in retrieving the filesystem attributes associated with the file that is being captured.

25 In a seventh development of the first aspect, the computer system is a virtual system and said first data storage device is a virtual disk.

An advantage is that the invention can be applied seamlessly to physical system or to virtual system.

In an eighth development of the first aspect, the handler presents said second data storage device as having a size at least equal to the size of said first data storage device.

An advantage is that the standard system capture mechanism such as back up and restore can be used.

5 In a ninth development of the first aspect, the method comprises the further steps of:

- receiving the master boot record data from said first data storage device; and
- sending said master boot record data to said handler so that said master boot record data is written on said second data storage.

10 Advantage is that all system data relevant for running the operating system can be captured.

In a tenth development of the first aspect, the method comprises the further steps of:

- receiving a list of excluded files which should not be captured from said computer system;
 - verifying whether said file stored on said first data storage belongs to said list of excluded files;
 - if said file does not belong to the list of excluded file, starting the file capture process, and
- 15 otherwise not.

An advantage is that files which are not required for running the operating system, such as temporary files, can be excluded from the system capture thus optimizing the size of the captured image.

20 In an eleventh development of the first aspect, the file data and said filesystem attributes are received from said second data storage device and sent to said first data storage device.

An advantage is that there is no use of a central repository, and replication to several target environments can happen at the same time while keeping the load on the source environment low.

25 According to a second aspect of the present invention, there is provided an apparatus comprising means adapted for carrying out each step of the method according to the first aspect of the invention.

An advantage is that this apparatus can be obtained very easily, thus making the method easy to execute.

According to a third aspect of the present invention, there is provided a computer program comprising instructions for carrying out the steps of the method according to a first aspect of the invention when said computer program is executed on a computer.

5 An advantage is that the invention can easily be reproduced and run on different computer systems.

According to a fourth aspect of the present invention, there is provided a computer readable medium having encoded thereon a computer program according to the third aspect of the invention.

10 An advantage is that this medium can be used to easily install the method on various apparatus.

Further advantages of the present invention will become clear to the skilled person upon examination of the drawings and detailed description. It is intended that any additional advantages be incorporated therein.

15 **Brief description of the drawings**

Embodiments of the present invention will now be described by way of example with reference to the accompanying drawings in which like references denote similar elements, and in which:

20 Figure 1 shows an environment in which an implementation of the present invention can operate.

Figure 2 shows a high-level process for capturing an image from a source environment and replicating it to a target environment.

Figure 3 shows a logical view of the elements from the source stack which get replicated it to the target stack.

25 Figure 4 shows a detailed view of the simulated disk or pseudo-disk.

Figure 5 shows an implementation of the present invention for replicating a source image to multiple target data storages.

Detailed description of the preferred embodiment

Figure 1 shows a system comprising:

- a source environment (100), comprising:
 - a source data storage (110),
 - 5 - an operating system (120) running on the source environment,
 - a deployed operating system (150),
 - a capturing agent (160) running on said deployed operating system (150),
 - a pseudo disk (170),
 - a backup restore tool (180) provided by the deployed operating system (150); and
- 10 - a target environment (130), comprising a target data storage (140).

The source data storage (110) comprises data which must be captured and transferred to the target data storage (140). The deployed operating system (150) is deployed on the source system by a component not represented in figure 1. It comprises the OS backup / restore tool (180) and the capturing agent (160). The pseudo disk (170) points towards the target data storage (140) on which the data must be replicated.

The source environment (100) as it is stored in the source data storage (110) must be replicated to a target system. The target environment (130) can be a computer system on which the source system must run, or a repository of computer environment images which will be later deployed to other computer systems on which these images will run. The target environment can be local to the source environment (100), such as a USB disk or a local hard disk. It can be a remote system connected via a network to the source environment (100). The network can be as simple as an Ethernet cable with the source and the target environments being in the same LAN, or the two environments can be very far apart.

The source environment (100) and the target environment (130) can be virtual or physical machines, or any combination of both.

A copy of the source data storage (110) block by block onto the target data storage (140) is inefficient, because empty blocks can be copied, the data can be fragmented and temporary or unused files can be copied as well. This results in an inefficient use of the bandwidth between the source and target environments, and in the waste of storage resources in the target data storage (140). In addition the capture of temporary data can be a privacy and security problem. This method does not work with physical machines.

Thus the data from the source data storage (110) must be retrieved at the filesystem level for an optimized replication. This way, useless files can be excluded from the capture. Furthermore

files won't be fragmented as they are written on the target data storage resulting in an optimized use of the storage resources of the target data storage (140). The source environment (100) must thus be running when the capture occurs. To avoid locking of files of the operating system (120) to be captured, the source environment must be booted on another operating system. This
5 operating system will be typically deployed from the network or from a local CD. There exist several operating systems (150) suitable for being deployed on various environments. For example, on Microsoft Windows WinPE2 can be used, and on Linux distributions such as LTSP can be used. Such OS can be booted from the network using for instance the PXE protocol. The choice of the operating system to be deployed (150) depends on the type of the operating system
10 (120) on the source environment (100). The deployed operating system (150) has two main purposes in the context of the invention: decode the filesystem used in the operating system (120) of the source environment (100), and provide an application which is able to retrieve data associated with a file but also the filesystem attributes associated with the file. Such file system attributes are indeed required for operating system to operate. Once the deployed operating
15 system (150) has been fully loaded into the source environment (100) memory, the capturing agent (160) is started. The capture agent (160) then mounts a simulated disk or pseudo disk (170) pointing towards the data storage disk (140) of the target environment (130). The location of the target disk can be provided for instance as a boot parameter to the deployed operating system (150). The pseudo disk (170) makes the target data storage (140) look like a local disk. Data can
20 be read from it and written to it. The target environment is thus no knowledge of the source filesystem format. In a preferred embodiment, the pseudo disk (170) presented size is at least equal to the one of the source data storage. The pseudo disk (170) will be described in more details with respect to figure 4.

Based on a description of the data to capture, such as a file list, the capture agent (160) will
25 leverage operating system applications to retrieve this data. In a preferred embodiment, the list of excluded file is provided so that these files are not captured. Such files are typically files with a .TMP extension, or Pagefile.sys or Hyberfil.sys on Microsoft Windows operating system. The description can also indicate disk partitions to include or exclude. For example the swap partition in a Linux environment would be typically excluded. In a preferred embodiment the description
30 of the data to capture comprises the master boot record and boot sectors data. The applications must be able to capture not only file data but also file metadata such as the filesystem attributes. In a preferred embodiment the application is the backup and restore application (180) associated with the operating system. In general the standard copy application provided by an operating system does not capture filesystem attributes. In Microsoft Windows environment, the

BackupWrite or BackupRead API or VSS can be used. In Linux environments the CPIO with the Fuse API can be used. The data is backed up in chunks to a temporary location such as a memory buffer.

The capture agent (160) then instructs the restore application (180) associated with the operating system to write the data from the memory buffer to the pseudo disk (170). Optionally the data can be compressed before it is written to the pseudo disk (170).

The output of the capturing process is thus a system image, with an optimized size, defragmented, which can be reused on other system with minimal further configuration, such as the IP address, the hostname, the product license keys etc.

Optionally the capture agent (160) can format the target data storage (140) through the pseudo disk (170) by means of the deployed operating system (150), so that it matches the filesystem format of the source data storage (110). Furthermore, after the filesystem format, the partition can be aligned to the block size, which is preferably 2MB.

Alternatively, the capture agent (160) and the pseudo disk could be deployed on the target environment; along with the deployed operating system (150) and the backup restore tool (180) if a system reboot is needed. The same principles described above could be applied to that configuration.

The choice of placing the capture agent on the source (100) or on the target environment (130) can be based on various factors. Preferably, the capture agent (160) and all the other components needed for executing the image capture should be deployed on:

- the physical environment if the other one is virtual, to reduce the load on the hypervisor of the virtualized system – in that case, access to the target data storage (140) can be provided by the hypervisor rather than the virtual machine as the hypervisor has a better bandwidth (the hypervisor does not need to be able to decode the filesystem format of the image);
- the less powerful environment if the two environments are of the same type (both are physical or virtual).

Figure 2 shows a high-level process for capturing an image from a source environment and replicating it to a target environment, comprising the steps of:

- starting the disk capture process (200);
- receiving the type of the operating system on the source computer(210);
- selecting a bootable environment (220);
- deploying the bootable environment to the source computer (230);

- booting the selected environment on the source computer (240);
- starting the capture agent on the source environment (250);
- mounting a pseudo disk on the source computer pointing towards a remote disk (260);
- starting the backup from the source data storage to a memory buffer (270);
- 5 - restoring the data in the memory buffer towards the pseudo disk (280);
- writing a block of data on the target data storage (290).

After mounting the pseudo disk (170), the capture agent can instruct the operating system (150) to format the target storage device (140). Having the correct file format helps in keeping the filesystem attributes when replicating an environment.

- 10 Booting to another operating system is useful to avoid locking of some operating system files during the capture process. However this can be optional and the capture agent (160) can be started directly on the original operating system (120) of the source environment (100).

Figure 3 shows a logical view of the elements from the source stack which get replicated to the target stack, the source stack comprises:

- a disk (300),
- a disk driver (310),
- a filesystem (320),
- an operating system (330),
- 20 - applications (340) and data (345); and the target stack comprises:
- a remote disk (350), which could be accessed through a pseudo disk (355),
- a disk driver (360), a file system (370), and
- data which has been replicated from the source stack, including the operating system (380) which is a subset of the operating system (330) of the source stack, and the applications (390)
- 25 and data (395).

The replication could be limited to a particular set of folders, thus enabling to choose which application or data is replicated.

Figure 4 shows a detailed view of the simulated disk or pseudo-disk (170) architecture and of its connection with the other components, figure 4 showing:

- a target system (400) with a device driver (410) which can be accessed from a network (420);
- a pseudo disk (430), comprising a device handler (440) which can support various disk formats (450), such as VHD, VMDK, VDI, RAW, the pseudo disk being connected to the device driver

(410) of the target system (400) over the network (420);

- a disk driver (460), listened by the pseudo disk (430) by means of a device driver listener (470);
- a device listener (470) for forwarding the requests from the disk driver (460) to the device handler (440) of the pseudo disk (430); and

5 - a filesystem driver (480).

The device handler (440) provides simple APIs to read or write on the pseudo disk (430) and to get the size of the disk. Some or all of the requests received by the device handler (440) can be forwarded over the network (420) to the device driver (410) of the target system (400). The device handler can leverage some local cache to avoid network queries. Thus the request to get
10 the size of the disk can be served immediately without further query to the remote disk (140). In a preferred embodiment, the device handler (440) presents a size at least equal to the size of the source data storage (110). The cache can also be used to avoid sending / receiving duplicate blocks, and to avoid transmitting blocks already stored at the target data storage (140) by first submitting to the device driver (410) a SHA signature of the data block and sending the actual
15 data only if it is new to the device driver (410). This can result in large bandwidth savings. The device handler (440) can also leverage remote physical disks, for example by using a Network Block Device on Linux, or it can read / write sectors on basic physical disks format over http in an agnostic manner.

20 Figure 5 shows an implementation of the present invention for replicating a source image to multiple target data storages, comprising:

- a source server (500), comprising a source data storage (510) and a network device server (520) for enabling network access to the source data storage (510);
- a first client system (530) comprising a target data storage (540), a capture agent (550) and a
25 pseudo disk (560) mounted by the capture agent (550); and
- a second client system (570) similar to the first client system, comprising a target data storage (580), a capture agent (590) and a pseudo disk (595) mounted by the capture agent (590).

The capture agents (530, 590) and the pseudo disks (560, 595) can be installed on the operating system of their respective client computer systems (530, 570), or deployed as described with
30 respect to the steps 210 to 260 of figure 2.

Both pseudo disks (560, 595) point towards the network device server (520). In a preferred embodiment, the capture process is similar to the one described with respect to steps 270 to 290 of figure 2, except that the data is backup from the pseudo disks (560, 595) to a memory buffer,

and then restored to the target data storages (540, 595). All the optimisation principles described with respect to figures 1 and 4 can be applied to the configuration shown in figure 5. To further optimize the bandwidth usage, multicasting can be used to transfer the block of data from the network device server (520) to the different pseudo disks (560, 590). This configuration has the further advantage that no central server is required to replicate directly environments. This method allows image migration from a physical or virtual source to a physical or virtual target.

Another embodiment comprises a method and system for optimally capturing an image on a physical or virtual environment and transferring it to a target system, by leveraging the capabilities of the operating system residing on the source system to manipulate the filesystem of the source storage device, and by presenting the target data storage as a local simulated disk to the source environment.

The invention can take the form of an entirely hardware embodiment, an entirely software embodiment or an embodiment containing both hardware and software elements. In a preferred embodiment, the invention is implemented in software, which includes but is not limited to firmware, resident software, microcode, etc.

Furthermore, the invention can take the form of a computer program product accessible from a computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any instruction execution system. For the purposes of this description, a computer-usable or computer readable medium can be any apparatus that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

The medium can be an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system (or apparatus or device) or a propagation medium. Examples of a computer-readable medium include a semiconductor or solid state memory, magnetic tape, a removable computer diskette, a random access memory (RAM), a read-only memory (ROM), a rigid magnetic disk and an optical disk. Current examples of optical disks include compact disk – read only memory (CD-ROM), compact disk – read/write (CD-R/W) and DVD.

A data processing system suitable for storing and/or executing program code will include at least one processor coupled directly or indirectly to memory elements through a system bus. The memory elements can include local memory employed during actual execution of the program code, bulk storage, and cache memories which provide temporary storage of at least some program code in order to reduce the number of times code must be retrieved from bulk storage during execution.

Input/output or I/O devices (including but not limited to keyboards, displays, pointing devices, etc.) can be coupled to the system either directly or through intervening I/O controllers.

Network adapters may also be coupled to the system to enable the data processing system to become coupled to other data processing systems or remote printers or storage devices through intervening private or public networks. Modems, cable modem and Ethernet cards are just a few of the currently available types of network adapters.

Claims

1. A method for capturing a file stored on a first data storage device of a computer system, said file being encoded in a first filesystem format on said first data storage device, said computer system comprising a first operating system with a first application suitable for
5 retrieving all the filesystem attributes associated with said file, said method comprising the steps of:
 - receiving a first message indicating a locator of a second data storage device;
 - creating on said computer system a handler to said second data storage device, so that said second data storage appears as a local device to said computer system;
 - 10 - instructing said first application to retrieve the file data and all filesystem attributes associated with said file from said first data storage;
 - instructing a second application to send said retrieved file data and said retrieved filesystem attributes to said handler so that said file data and said filesystem attributes are written on said second data storage.
- 15 2. The method of claim 1 comprising the further step of compressing the file data before sending it to said handler.
3. The method of claims 1 or 2 further comprising the prior steps of:
 - receiving a message identifying the type of said first operating system;
 - deploying an agent for running a file capture on a second operating system which is of the same
20 type as said first operating system so that it can manage said first filesystem format;
 - booting said second operating system on said computer system, so that said agent executes the steps of the claims 1 or 2.
4. The method of any of the preceding claims wherein said second storage device is a remote disk.
- 25 5. The method of any of the preceding claims further comprising the step of formatting said second storage location so that its filesystem format is the same as said first filesystem format.
6. The method of claim 5 wherein a partition is created on said second storage location and said partition is aligned on the block size of said first filesystem format.
7. The method of any of the preceding claims wherein the steps of receiving and sending are
30 respectively performed by a backup and restore application associated with the operating system.

8. The method of any of the preceding claims wherein said computer system is a virtual system and said first data storage device is a virtual disk.

9. The method of any of the preceding claims wherein said handler presents said second data storage device as having a size at least equal to the size of said first data storage device.

5 10. The method of any of the preceding claims comprising the further steps of:
- receiving the master boot record and boot sectors data from said first data storage device; and
- sending said master boot record and boot sectors data to said handler so that said master boot record and boot sectors data is written on said second data storage.

10 11. The method of any of the preceding claims comprising the further steps of:
- receiving a list of excluded files which should not be captured from said computer system;
- verifying whether said file stored on said first data storage belongs to said list of excluded files;
- if said file does not belong to the list of excluded file, starting the file capture process, and otherwise not.

15 12. The method of any of the preceding claims wherein said file data and said filesystem attributes are received from said second data storage device and sent to said first data storage device.

13. An apparatus comprising means adapted for carrying out each step of the method according to any one of the claims 1 to 12.

20 14. A computer program comprising instructions for carrying out the steps of the method according to any one of claims 1 to 12 when said computer program is executed on a computer.

15. A computer readable medium having encoded thereon a computer program according to claim 14.

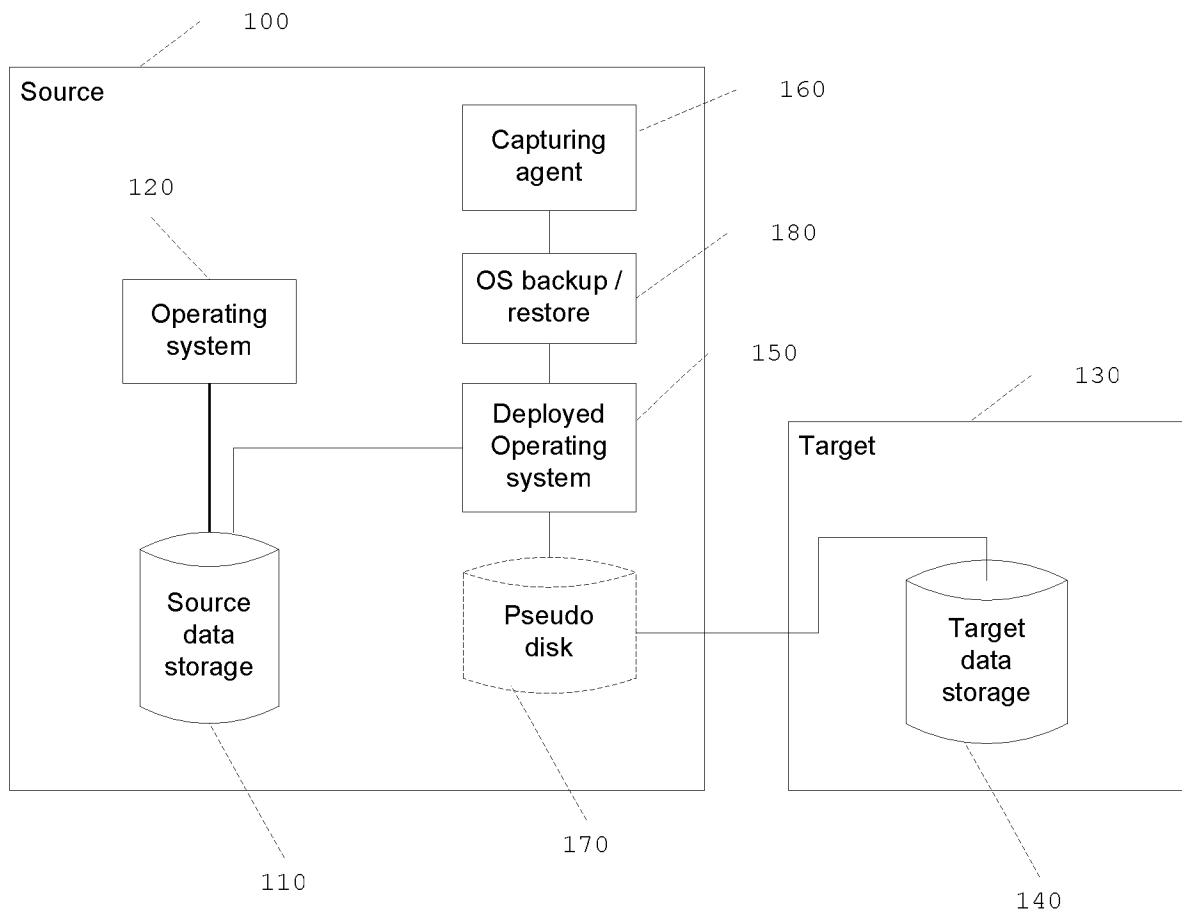


Figure 1

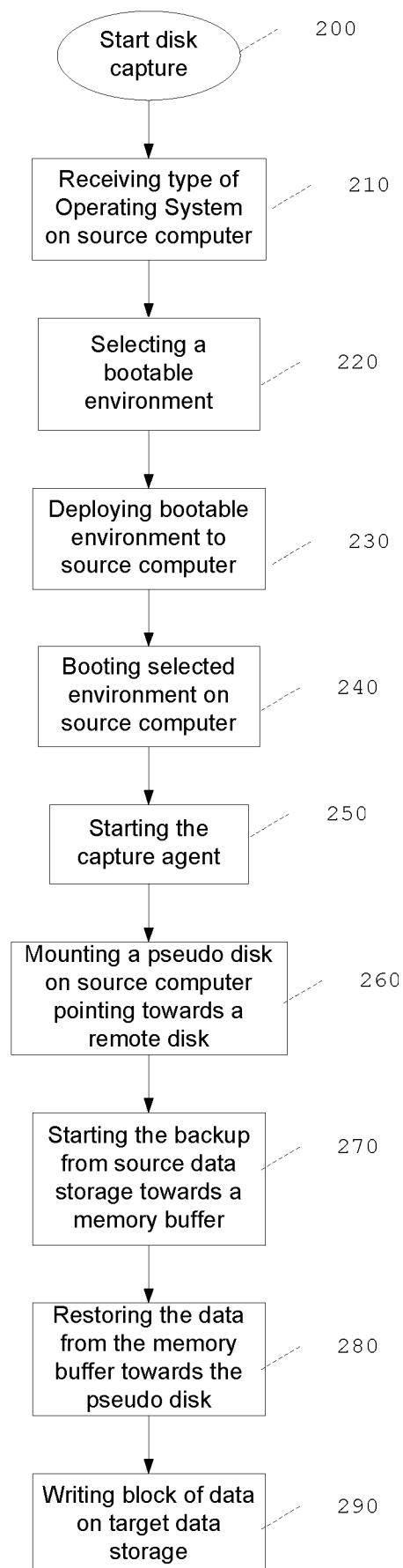
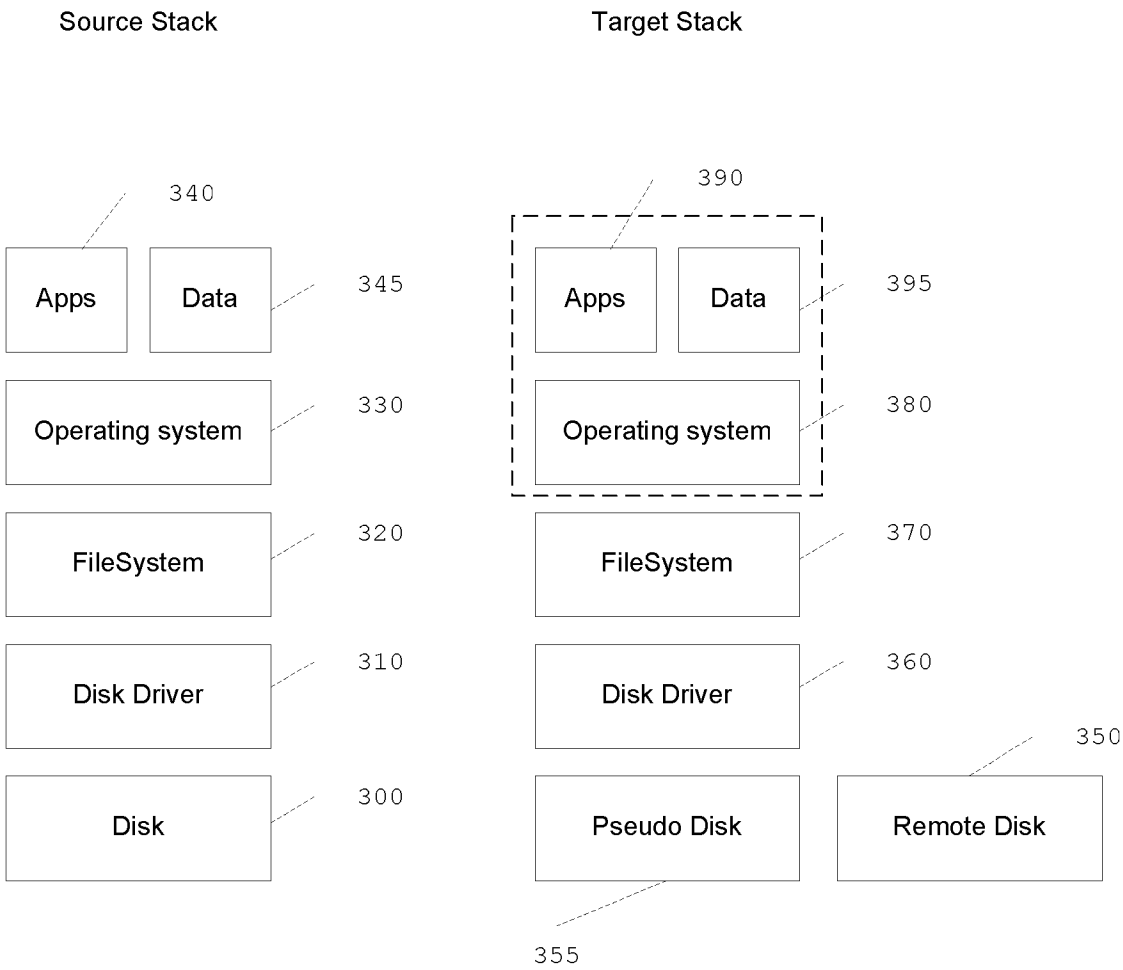


Figure 2



150

Figure 3

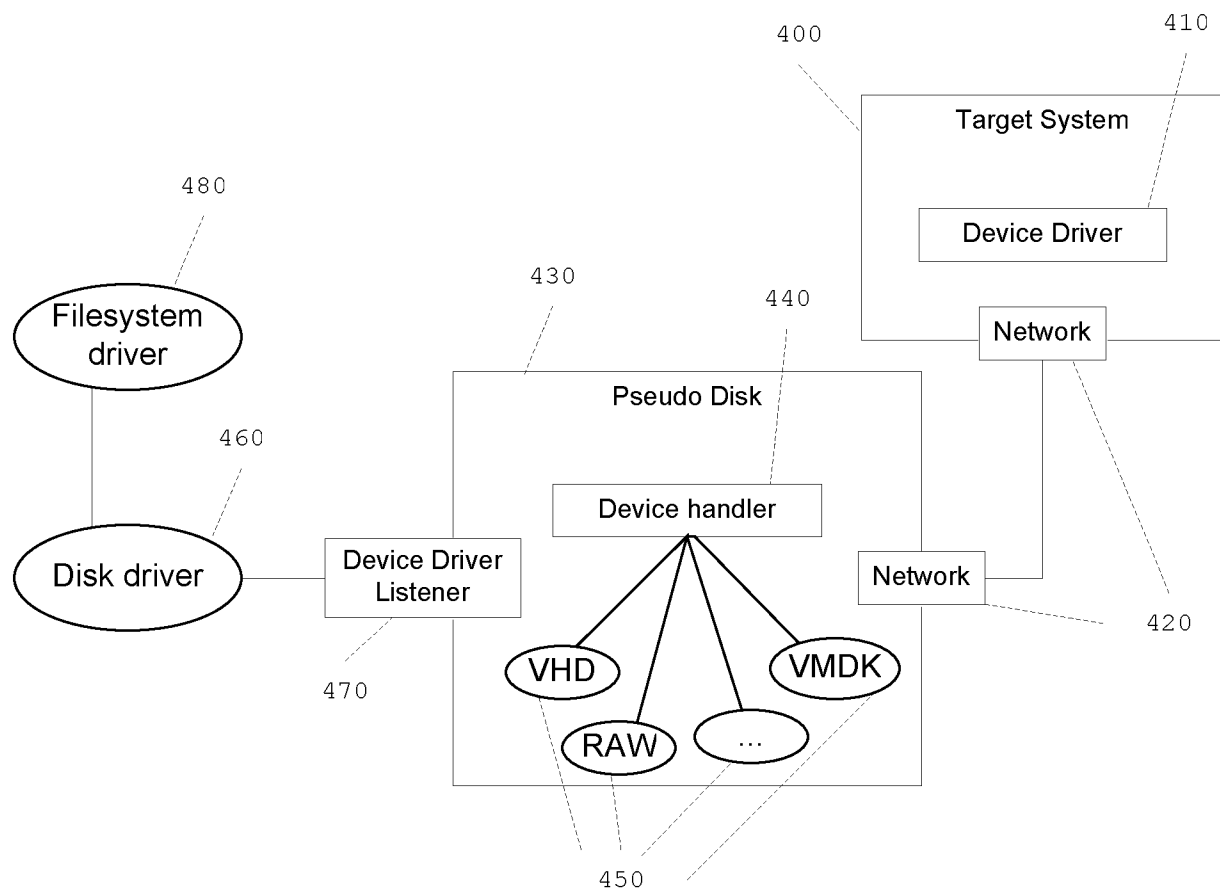


Figure 4

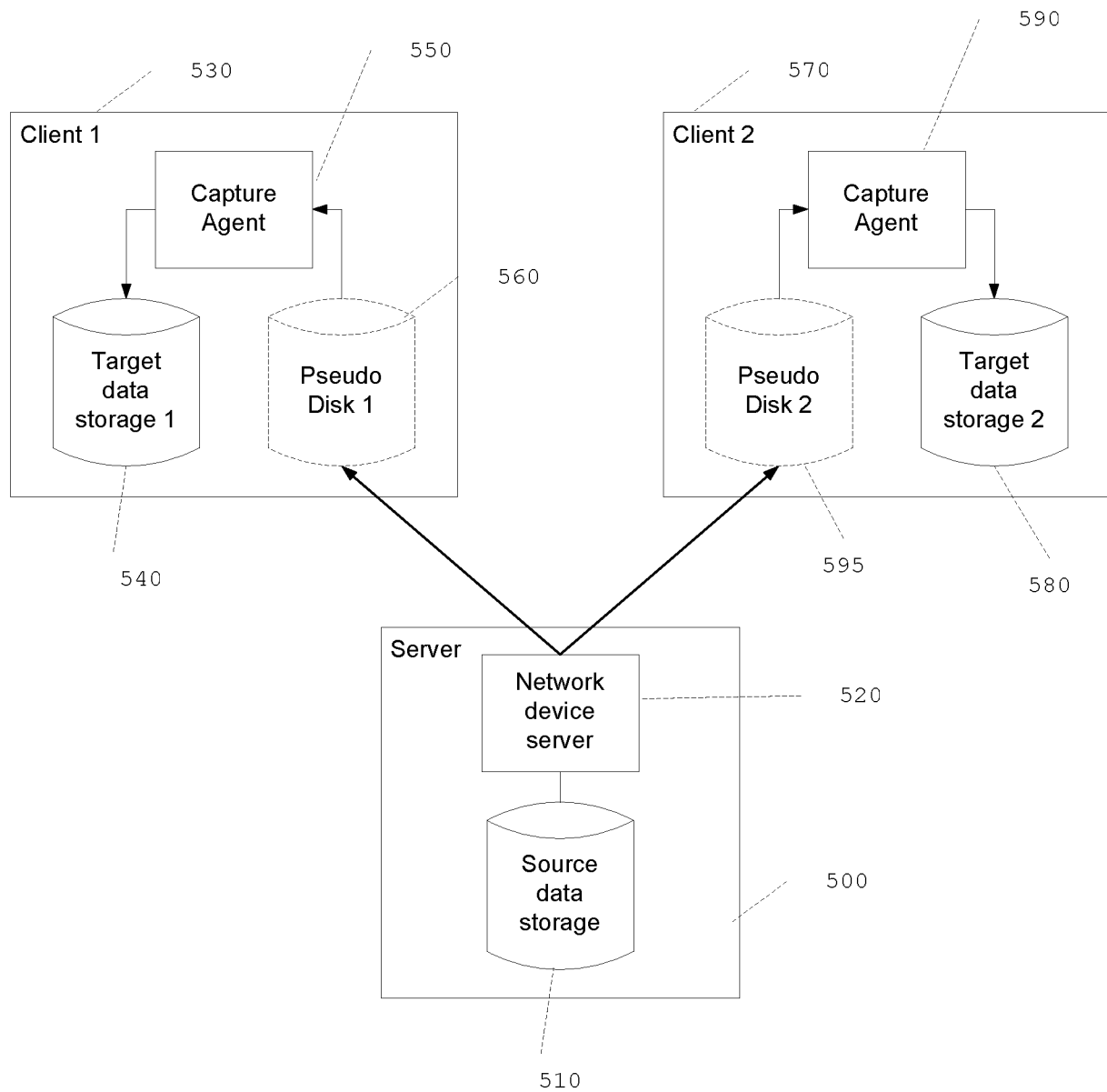


Figure 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/055833

A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F11/14

ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, INSPEC, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5 832 274 A (CUTLER BRUCE R [US] ET AL) 3 November 1998 (1998-11-03) column 3, line 1 - line 27 column 3, line 40 - column 5, line 19 column 5, line 52 - column 6, line 17 column 6, line 66 - column 7, line 51 column 9, line 7 - column 10, line 41 column 11, line 25 - column 12, line 17 figures 1,2	1-15
A	US 2003/154220 A1 (CANNON DAVID MAXWELL [US]) 14 August 2003 (2003-08-14) paragraphs [0011], [0024] - [0027], [0053], [0065], [0066], [0083] - [0089], [0093] - [0112] figures 1a,1b,2,6,7	1-15

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

24 June 2010

Date of mailing of the international search report

30/06/2010

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Johansson, Ulf

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2010/055833

Patent document cited in search report		Publication date	Patent family member(s)	Publication date
US 5832274	A	03-11-1998	NONE	
US 2003154220	A1	14-08-2003	US 2005146945 A1	07-07-2005