DOI https://doi.org/10.15407/usim.2019.04.042
UDC 004.9:004.75

YU.M. LYSETSKYI, Dr. of Eng. Sci., General Director, DP «S&T UKRAINE»,
Prosp. Akad. Palladina, 44, 03680, Kiev, Ukraine,
Yuri.Lysetskyi@snt.ua

YE.P. SEREDOVICH, Senior Engineer for Platforms and
Operating Systems Support, DP «S&T UKRAINE»,
Prosp. Akad. Palladina, 44, 03680, Kiev, Ukraine,
Yevgen.Seredovich@snt.ua

SOFTWARE-DEFINED STORAGES AS A TOOL OF LOWERING IT INFRASTRUCTURE COSTS

The paper considers software-defined storages, their essential requirements, features, as well as advantages and disadvantages of solutions by the different vendors. The most widely spread OpenSource products used to build software-defined storages are analysed. An example of functioning software-defined storage based on non-commercial software Gluster and decommissioned hardware is given. The paper describes storage architecture which provides the reliable storing of information with minimal costs incurred.

Keywords: *software-defines storages, opensource, SDS, software, backup system, virtualization, volume, scalability, architecture.*

Introduction

Costs connected with storages can make up to 25% of total IT infrastructure costs in large organizations. This number can grow even more in case of data volume growth and increasing demand for storage capacity [1]. On the other hand, organizations have limited IT budgets which makes necessary to search for technological solutions both lowering costs for storage implementation and preserving service quality.

One of such solutions are software-defined storages or SDS which enable implementation of storages based on commodity equipment. As a rule, it can be a group of server nodes with x86-64 architecture running generic operating systems (OS) such as Linux, Windows, FreeBSD. Basic dis-tinctive feature of SDS is the virtualization of storage function which separates hardware from software managing the storage infrastructure [2].

SDS is one of those new technologies that break into the IT infrastructure of organizations and cloud service providers. According to IDC forecast global annual expenses on software-defined storages will grow by approximately 13,5% between 2017 and 2021 to reach $16,2 billion [3]. Gartner expects that 70–80% of unstructured data will be stored in inexpensive storages under SDS control by 2020 and 70% of existing massive storages will be available as purely software in 2019.

SDS Requirements and Features

Basically SDS is a software managing storage services based on business oriented policies that are independent on equipment. Different SDS definitions usually include storage virtualization to separate hardware from software managing storage infrastructure. Software including a software-defined storage environment is to provide an elastic storage organization as well as policy based management of such functions as cashing, deduplication, snapshots, backing up, thin provisioning.

Storage Networking Industry Association (SNIA) defines SDS as a virtualized storage environment featuring the interface for storage management and including the following [4]:

▪ Automation — simplified management that reduces the cost of maintaining storage infrastructure.

▪ Standard Interfaces — APIs for the management, provisioning and maintenance of storage devices and services.

▪ Data Path — block, file and/or object interfaces that support applications written to these interfaces.

▪ Scalability — seamless ability to scale the storage infrastructure without disruption to the specified availability or performance.

▪ Transparency — the ability for storage consumers to monitor and manage their own storage consumption against available resources and costs.

Quite a lot vendors offer SDS today: Dell EMC (Dell Nexenta, EMC and ScaleIO), HPE (StoreVirtual VSA), IBM (Spectrum Storage), NetApp (ONTAP Select), VMware (vSAN), Red Hat (RedHat Ceph Storage и RedHat Gluster Storage), StoneFly (SCVM, SDUS), DataCore (SANsymphony), VMware (vSAN), Virtuozzo (vStorage), Microsoft (Storage Spaces Direct) etc. All SDS can be grouped into three categories:

▪ classical (CEPH, Red Hat Storage Server, EMC ScaleIO);

▪ based on traditional storages (NetApp ONTAP Select, HPE StoreVirtual VSA);

▪ integral part of complexes (VMware vSAN).

Some manufacturers offer both complex solutions and software component (Huawei, Dell EMC). It allows for more elastic product selection and usage of outdated hardware to address less demanding storage tasks. Another feature of SDS is the ability to use disk array virtualization on some classical storages.

SDS can have scale-out or distributed (with no common elements) architectures.

In the former case the redundancy is achieved by means of distributed data copies, however, the redundancy of communications between nodes lowers write speed. Data transfer network is a critical element so such solutions are usually implemented on InfiniBand basis. This principle is used by SDS VMware vSAN, HPE StoreVirtual VSA, Dell EMC ScaleIO.

In systems with no common elements the data are written to one node and then copied to other nodes by schedule to provide redundancy. Such write operations to storage are not transactional, so such an approach is more economical. Ethernet is mostly used as an interconnect and this architecture is quite convenient for scalability.

Today SDS are in demand in the IT market as their implementation offers a number of advantages including: abstraction from the lower level of hardware, scalability, simplification of storage infrastructure and relatively low cost.

SDS Implementation Case

Ukrainian office of Endress+Hauser, a prominent Swiss manufacturer of industrial control systems components hired systems integrator S&T Ukraine to solve the issue of lacking storage resources. The task had to be addressed by way of implementation of additional storage based on available decommissioned hardware, with minimal changes to existing IT infrastructure and with minimal costs.

Integrator offered implementation of software-defined storage based on non-commercial products. After customer's requirements analysis there were chosen Ceph and Gluster products.

Ceph is a distributed storage with high scalability and performance with no single point of failure. It was initially developed for scalability to Exabyte levels and higher with utilization of commodity platforms. It is not only an unlimited scalability and productivity that Ceph offers but also performance and elasticity. This way it removes the need for expensive storages. Ceph is a defined software and unified solution for enterprise level storages working on standard hardware. This makes it the most cost effective and multifunctional storage. Universal Ceph storage provides both file and block access interfaces. It enables elastic usage of data storage based on customer requirements. To use Ceph FS it is enough to have metadata server at any of the Ceph cluster nodes. However, you have to keep in mind that availability of only one MDS server creates a single point of failure. It is recom-

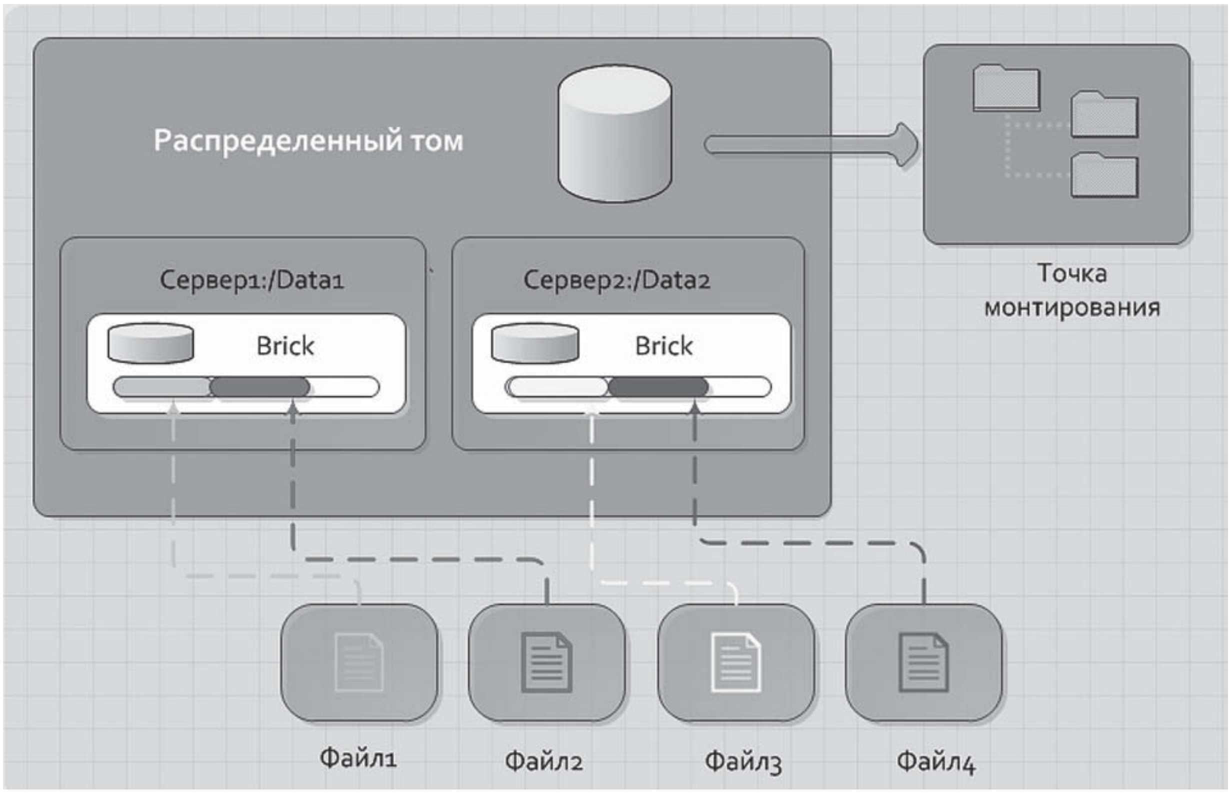

Fig. 1. Distributed volume implementation

mended that the cluster should have at least two separate networks, the first being front-side (public network) and the second being back-side (cluster network). Two networks are recommended to separate customer data and Ceph cluster data exchange. If both networks are deployed in one physical environment there can arise productivity issues. These separate networks should have a capacity of at least 1 Gbps, however 10 Gbps networks are recommended.

Gluster is a distributed scalable file system with a feature of quick allocation of additional disk space to meet user demands. The Gluster clusters together its building blocks over Infiniband RDMA or TCP/IP interconnect (aggregating memory and disk resources) and manage data in a single global namespace. Automatic failover is declared as one of key functions. This solution also does not require a centralized metadata server, as the main building principle is clustering of resources. And the resources are usually the hardware already decommissioned by the customer.

Of all considered variants the former did not quite meet the customer requirements as demanded additional expenses on upgrade of existing network infrastructure. This is why it was decided to deploy additional storage based on Gluster with functionality equal to that of Ceph but having more flexible requirements to IT infrastructure.

In our case Gluster unites disk resources into single volumes. A volume can be created from bricks (disks) of one or different servers.

There can be following volume types [5]:

Distributed. Distributed volumes distribute files randomly across the bricks in the volume. This option is used when there is a need of maximal scaling of available storage. Redundancy is not a priority in this case or provided by other software or hardware levels. Example, of implementation see in Fig. 1.

Replicated volume (RAID-1 analogue). The files are replicated across numerous bricks in the volume. Replicated volumes are used in environments where high availability and high reliability are critical.

Striped volume (RAID-0 analogue). The data are striped across bricks in the volume. Such volu-mes are used for environments with large numbers of concurrent file accesses. They are ideal for accessing very large files.

Dispersed volume (RAID-5/RAID-6 analogue). For this volume type a file is split into fragments to which the redundancy information is added and then fragments are successively written to volume. This approach is used when a certain set level of reliability is required with a minimal toll on available storage space. The reliability level can be set up manually, based on customer requirements.

This volume type has been selected as a basis for additional storage while decommissioned hardware has been clustered with aggregation of the disk space. The chosen reliability level was equal to that of RAID-5 allowing for failure of one block (a server or its component) without losing data on the whole cluster.

Scaling up enabled following variants:

▪ use as an independent file resource and allow users save documents to it;

▪ connect to existing servers as a block device;

▪ go for non-standard solution, like use as a back up system component.

In view of the fact that the customer did not have a fully functional back up system and backing up was done by way of replicating data between servers, the decision was made in favor of this option. As the budget was strictly limited the back up system was implemented on the basis of OpenSourcs products. Based on product analysis there were identified three variants for Open-Source back up system that met customer requirement. They were Amanda (Zmanda), a free version of Amanda Community Edition; Bacula, a free version of Community edition and Bareos a purely OpenSource product, an offshoot of Bacula project.

Bacula creators have been working on closing the source code since 2010 and now they make changes only to the closed (Enterprise) source, do not accept third party patches and cut functionality of the open code. In view of these facts going for this product has not been considered expedient. Amanda is less functional compared to Bacula product and possesses rather limited capabilities of integration with SDS. On the other hand, Bareos features good specifications and its developers position the product as a fully functional back up system.

In view of the above and the fact that Bareos features good capability of integration with Gluster and provides for creation of backup copies from and to clustered resources, the decision was made in favor of choosing this product.

Thus, there emerged the opportunity to build a distributed and scaled storage fully meeting customer requirements. Central backup server (or Director as per Bareos classification) has been deployed as a virtual machine at one of virtualization hosts. Bareos proved to be quite elastic in setup and allowing for connection of additional modules and plugins. The only drawback was that all settings had to be done manually in respective configuration files due to absence of graphic interface for system setup and administration. Additional plugin enabled connection of a united disk storage, an ear-

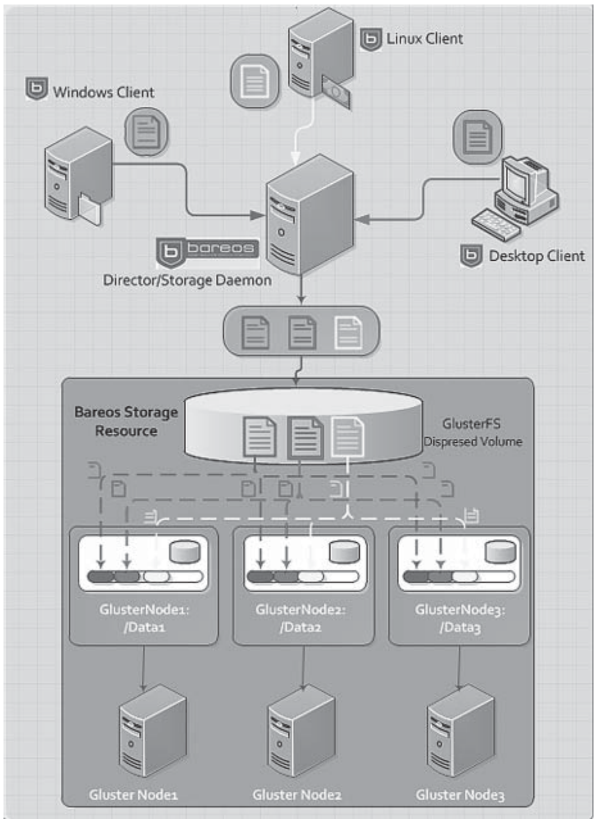

***Fig.* 2.** Architecture of software-defined storage lier configured cluster, to the server. Ordinary personal computers were used as cluster blocks. The client of the back up system was deployed on servers and workstations (Bareos clients are available for majority of operating systems). Back up copies are stored to cluster building blocks (server nodes). In case one node fails the whole node or its components are replaced. If there is insufficient storage space the next node is added to expand the volume. Solutions architecture is presented in Fig. 2.

This storage has been successfully implemented, tested and commissioned.

Two open independent OpenSource products were used to implement a fully functional complex solution for backing up which required no significant expenses. Some components of the solution can be used independently or as storages additional to already used by the customer. Thus the integrator was able to address simultaneously a number of issues with customer's IT infrastructure and meet the demands made during initial stage.

Conclusion

The article considers software-defined storages, their essential requirements, features, as well as advantages and disadvantages of solutions by different vendors. The most widely spread Open-Source products used to build software-defined storages are analyzed.

An example of functioning software-defined storage based on non-commercial software Gluster and decommissioned hardware is given. The article describes storage architecture which provides reliable storing of information with minimal costs incurred. Therefore, the SDS technologies can significantly reduce the costs of storages and their administration. And it is for these reasons that many companies are actively developing software (Atlantis Computing, Maxta, StarWind, DataCore Software, Sanbolic, Nexenta, CloudByte) and complex SDS solutions (Dell EMC, IBM), as well as specialized devices (Tintri, Nimble, Solidfire).

REFERENCES

1. *Lysetskyi, Yu.M.*, 2013. “Korporativnye sistemy hranenija dannyh. Primer postroenija”. [Corporate Storages. Implementation Case]. Upr. sist. mas., 6, pp. 68–71 (In Russian).
2. *Ivanov, I.K.* Programmno-opredeljaemye SHD: dostupnoe reshenie dlja nedostupnyh zadach [Software-Defines Storages: an Available Solution for Unapproachable Tasks]. [online] Available at: <http://www.globalcio.ru/workshops/1760/> [Accessed 1 Feb. 2019]. (In Russian).
3. *Rynok programmno-opredeljaemyh* hranilishh budet rasti ezhegodno na 13,5% [Software-Defined Storages Maket will Grow at 13,5% Annually]. [online] Available at: <http://www.dailycomm.ru/m/41316> [Accessed 10 Apr. 2019].
4. *Programmno-opredeljaemye SHD*: sravnivaem 7 reshenij [Software-Defined Storages: Comaping 7 Solutions] . [online] Available at: <https://habr.com/ru/company/lanit/blog/324072/> [Accessed 3 May 2019].
5. *Gluster FS Documentation*. [online] Available at: <http://gluster.readthedocs.io/en/latest/> [Accessed 13 Apr. 2018].

Received 05.08.2019

ЛІТЕРАТУРА

1. *Лисецкий Ю.М.* Корпоративные системы хранения данных. Построение. УСиМ, 2013, 6, С. 68–71.
2. *Иванов И.К.* Программно-определяемые СХД: доступное решение для недоступных задач. [online], http://www.globalcio.ru/workshops/1760/.
3. *Рынок программно-определяемых* хранилищ будет расти ежегодно на 13,5%. [online], Available at: http://www.dailycomm.ru/m/41316.
4. *Программно-определяемые СХД*: сравниваем 7 решений. [online], https://habr.com/ru/company/lanit/blog/324072/.
5. *Gluster FS Documentation*. [online], http://gluster.readthedocs.io/en/latest/.

Надійшла 05.08.2019

Ю.М. Лисецький, д-р техн. наук, генеральний директор,
ДП «ЭС ЭНД ТИ УКРАИНА»,
Київ, 03680, просп. Академіка Палладіна, 44, Україна,
Yuri.Lysetskyi@snt.ua

Є.П.Середович, ст. інженер з обслуговування платформ і операційних систем,
ДП «ЭС ЭНД ТИ УКРАИНА»,
Київ, 03680, просп. Академіка Палладіна, 44, Україна,
yevgen.seredovich@snt.ua

СИСТЕМИ ЗБЕРЕЖЕННЯ ДАНИХ, ЯКІ ВИЗНАЧАЮТЬСЯ ПРОГРАМНО, ЯК ІНСТРУМЕНТ СКОРОЧЕННЯ ВАРТОСТІ ІТ-ІНФРАСТРУКТУРИ

Вступ. У великих організаціях вартість систем збереження даних (СЗД) складає до 25 відсотків вартості ІТ-інфраструктури, тому завдання пошуку технологічних рішень, що дозволяють скорочувати витрати на їх створення, є актуальним. Одне з них — СЗД, які визначаються програмно (*software-defined storage – SDS*), що дозволяють створювати сховища даних на неспеціалізованому обладнанні масового класу, під керуванням операційних систем загального призначення.

Вимоги до *SDS* та їх особливості. *SDS* визначають як віртуалізоване середовище збереження даних з інтерфейсом ке-рування сервісами, яке повинно включати в себе: автоматизацію управління; стандартні інтерфейси – *API*; віртуалізацію шляхів доступу до даних; масштабованість; моніторинг ресурсів, що споживаються. *SDS* можна розділити на три категорії: класичні; на основі традиційних систем збереження; у складі обчислювальних комплексів.

Приклад реалізації *SDS*. Для вирішення завдання збільшення ресурсів для збереження інформації необхідно було створити додаткову СЗД з мінімальними фінансовими витратами. Було запропоновано побудувати СЗД, яка визначається програмно, з використанням програмних продуктів та комп'ютерної техніки, виведеної з експлуатації. За результатами аналізу було обрано продукт *Gluster*. Враховуючи те, що замовник не мав повноцінної системи резервного копіювання (СРК), її було реалізовано на базі *OpenSource* продукту *Bareos*, що дозволило не тільки створювати резервні копії, але й зберігати їх на кластерних ресурсах. Це надало можливість створення розподіленої та масштабованої СЗД, яка повністю відповідає вимогам замовника. Центральний сервер резервного копіювання було розгорнуто у вигляді віртуальної машини, на одному з хостів віртуалізації. За допомогою додатково встановленого плагіна до сервера було підключено єдине дискове сховище — кластер, який було налаштовано раніше. Як складові блоки кластеру було використано звичайні персональні комп'ютери. На серверах та робочих станціях було розгорнуто клієнтську частину СЗД, а резервні копії зберігалися на серверних нодах кластеру. При виході з ладу однієї ноди замінюється або одна нода, або її компоненти. Якщо не вистачає місця для збереження даних, тоді додають наступну ноду.

Таким чином, з двох відкритих незалежних продуктів було створено повноцінне комплексне рішення з резервного копіювання даних, без суттєвих витрат.

Висновок. У статті розглянуто основні вимоги до СЗД, що визначаються програмно, а також їх особливості. Проведено аналіз найбільш поширених некомерційних *SDS*, показано їх переваги та недоліки. Описано приклад розробки і реалізації *SDS* на базі *OpenSource* продуктів і невикористаної обчислювальної техніки, виведеної з експлуатації, що дозволяє суттєво знизити вартість СЗД.

Ключові слова: *системи зберігання даних, що визначаються програмно, OpenSource, SDS, системи резервного копіювання, віртуалізація, обсяг, масштабованість, архітектура.*

Ю.М. Лисецкий, доктор технических наук, генеральный директор,
ДП «ЭС ЭНД ТИ УКРАИНА»,
просп. Академика Палладина, 44, Киев, 03680, Украина,
Yuri.Lysetskyi@snt.ua

Е.П. Середович, ст. инженер по обслуживанию платформ и операционных систем,
ДП «ЭС ЭНД ТИ УКРАИНА»,
просп. Академика Палладина, 44, Киев, 03680, Украина,
Yevgen.seredovich@snt.ua

ПРОГРАММНО-ОПРЕДЕЛЯЕМЫЕ СИСТЕМЫ ХРАНЕНИЯ ДАННЫХ КАК ИНСТРУМЕНТ СОКРАЩЕНИЯ СТОИМОСТИ ИТ-ИНФРАСТРУКТУРЫ

Введение. В крупных организациях стоимость систем хранения данных (СХД) составляет до 25 процентов стоимости ИТ-инфраструктуры, поэтому задача поиска технологических решений, позволяющих сокращать расходы на их создание, — актуальна. Одно из решений — программно-определяемые СХД (*software-defined storage — SDS*), позволяющие создавать хранилища данных на неспециализированном оборудовании массового класса, под управлением операционных систем общего назначения.

Требования к *SDS* и их особенности. *SDS* определяют как виртуализированную среду хранения данных с интерфейсом управления сервисами, которая должна включать в себя: автоматизацию управления; стандартные интерфейсы — *API*; виртуализацию путей доступа к данным; масштабируемость; мониторинг потребляемых ресурсов. *SDS* можно разделить на три категории: классические; на основе традиционных систем хранения; в составе вычислительных комплексов.

Пример реализации *SDS*. Решая задачи увеличения ресурсов для хранения информации необходимо создать дополнительную СХД при минимальных финансовых затратах. Было предложено построить программно-определяемую СХД, используя некоммерческие программные продукты и компьютерную технику, выведенную из эксплуатации. В результате анализа был выбран продукт *Gluster*. Учитывая, что у заказчика не было полноценной системы резервного копирования (СРК), ее реализовали на базе *OpenSource* продукта *Bareos*, что позволило не только создавать резервные копии, но и хранить их на кластерных ресурсах. Это дало возможность создания распределенной и масштабируемой СХД, полностью отвечающей требованиям заказчика. Центральный сервер резервного копирования был развернут в виде виртуальной машины, на одном из хостов виртуализации. При помощи дополнительно установленного плагина, к серверу подключено единое дисковое хранилище — сконфигурированный ранее кластер. В качестве составных блоков кластера используются обычные персональные компьютеры. На серверах и рабочих станциях — развернута клиентская часть СРК, а резервные копии сохраняются на серверных нодах кластера. При выходе из строя одной ноды — заменяется вся нода, либо ее компоненты. Если недостаточно места для хранения данных, то добавляется следующая нода. Таким образом, из двух открытых независимых продуктов было создано полноценное комплексное решение по резервному копированию данных, без каких-либо существенных капиталовложений.

Заключение. В статье рассмотрены основные требования к программно-определяемым СХД, а также их особенности. Проведен анализ наиболее распространенных некоммерческих *SDS*, показаны их преимущества и недостатки. Описан пример разработки и реализации *SDS* на базе *OpenSource* продуктов и неиспользуемой вычислительной техники, что позволяет значительно снизить стоимость СХД.

Ключевые слова: *программно-определяемые системы хранения данных, OpenSource, SDS, системы резервного копирования, виртуализация, объем, масштабируемость, архитектура.*