OBJECT STORAGE CHARGEBACK - EMC ECS AND VIPR SRM

Brian Dehn
Principal Systems Engineer
EMC
Brian.Dehn@emc.com

EMC²

Table of Contents

Table of Figures

Table of Tables

Introduction

The popularity of object storage continues to grow as enterprises introduce web-scale applications, save money with new archive platforms, implement object storage as a service, introduce big data analytics, and more. However, object storage is new to many IT departments, and standard processes and tools are not sufficient to support operationalization, resulting in questions like: What is object storage? How is it different from traditional block and file storage? What is the architecture? How should I report on it? How should I charge back my customers for using it?

Object storage is used for many purposes and is a key storage platform for most public and private cloud solutions. In the public cloud arena, vendors such as Amazon, Google, and EMC offer cloud storage solutions. Cloud applications such as Facebook and Dropbox utilize object storage platforms for storing videos, pictures, spreadsheets, and other unstructured data. Cloud storage vendors such as EMC provide on-premises object storage solutions that enable enterprises to deploy their own private and hybrid cloud storage offerings.

In many of these scenarios, consumers must be charged for the storage they use. To address the topic of charging back for object storage, this article provides a short overview of object storage, differentiates it from more traditional types of storage, and introduces EMC Elastic Cloud Storage™ (ECS). The article then explores ECS architecture, concepts, and logical constructs including multi-tenancy, replication groups, namespaces, and buckets and relates them to reporting requirements with a focus on chargeback. Finally, it explains how EMC ViPR® SRM can be used to meet these reporting requirements through built-in reports and powerful customization features.

What Is Object Storage?

Before delving into the topic of charging back for object storage usage, a fundamental understanding of object storage and how it compares to other types of more traditional storage is required.

Note: This article presents an overview of object storage to establish a foundation upon which the discussion of chargeback will be built. For more in-depth discussions of object storage, many articles, papers, and blogs are available and can be found with simple web searches like 'what is object storage'.

Description of Object Storage

To define object storage, we must first define 'object'. An object is a unit of unstructured data such as a spreadsheet, picture, movie, document, etc., oftentimes combined with data describing the object. Building on that definition, object storage is a type of software-defined storage that abstracts underlying hardware and allows data to be managed at the object level.

Object storage vendors seek to provide solutions and capabilities not available with traditional block and file storage solutions, including:

- Programmable interfaces for applications
- Management of data at the object level
- Namespaces that span multiple globally-distributed sites and instances of hardware
- Object-level data protection and replication
- Total software independence from the underlying storage hardware
- Scalability to exabytes
- Architecture designed to support mobile, cloud, big data, and social networking applications

Applications typically communicate with an object storage platform using one or more common REST-based application programming interfaces (APIs), such as Amazon's Simple Storage Service (S3) API, OpenStack Swift API, EMC Elastic Cloud Storage (ECS) API, EMC Atmos API, EMC Centera API, and more. These APIs allow applications to manage storage of objects using basic CRUD (Create, Read, Update,

Delete) operations. Most object storage vendors also provide extensions to their APIs, providing unique and differentiating capabilities.

When stored in an object storage platform, each object usually consists of three components:

1. The object data itself
2. Descriptive data about the object, known as metadata
3. A globally unique object identifier (UID)

Metadata and UIDs provide advantages over other types of storage. Application developers have the option of including metadata with objects when they are created and/or updated. Metadata can be used for multiple purposes – for example, querying for objects based on search criteria, which eliminates the need to extract object data from the object storage platform for basic query operations. UIDs eliminate duplicate name conflicts and allow for namespaces and buckets of unlimited sizes.

These object storage capabilities and features enable reliability, availability, and scalability generally associated with cloud storage, and many cloud application vendors, service providers, and enterprises utilize object storage platforms. For example, in April 2009, Facebook published information about an object storage platform they developed called Haystack to store billions of photos uploaded by users.[1] Syncplicity, a leading secure file sharing and collaboration cloud solution, supports multiple back-end object storage platforms including EMC Elastic Cloud Storage (ECS) and EMC Atmos®. EMC announced plans in October 2015 to form a new Cloud Services business under the Virtustream brand, one component of which is EMC object storage services built on ECS.[2], a software-defined object storage platform that leverages low-cost commodity hardware and enables enterprises to deploy their own on-prem cloud storage offerings.

---

[1] Facebook. "Needle in a haystack: efficient storage of billions of photos." Facebook, 2009. Web. February 15, 2016. (Facebook 2009)

[2] EMC. "EMC Reveals New Cloud Services Business." EMC. October, 2015. Web. February 15, 2016. (EMC 2015)

One way to learn more about object storage is to play with it. To demonstrate a quick and easy way to play with object storage, a video was created as a supplement to this article. This video shows how you can use mobile devices, free cloud storage apps, and ECS Test Drive (EMC's free publically available object storage platform) to become familiar with object storage. To watch the video, double-click on the video image.

Object storage vendors have purposefully designed many of the features and capabilities mentioned here into their object storage solutions to address requirements of cloud-scale applications and provide capabilities not available with traditional block and file storage solutions.

How Does Object Storage Compare To Block & File Storage?

Object storage is different from traditional block and file storage in several ways, including architecture, scaling, performance, use cases, and chargeback. As a general statement, characteristics of object storage minimally overlap with block storage and partially overlap with file storage. Understanding the differences is important when considering how to implement chargeback for object storage.

Architecture

Object storage architecture is different from that of block and file storage arrays, and this section provides a review of traditional storage types with a focus on the differences as compared to object storage.

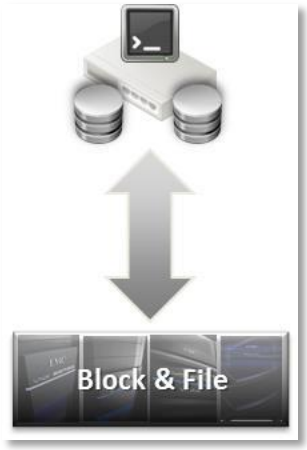


Figure 1 - Block & File

Block storage allows creation of devices (also known as volumes and LUNs) that consist of raw storage capacity, provide data protection with various RAID schemes, are allocated and mounted to server operating systems, and are used as if they are local physical disks. Common access protocols include Fiber Channel and iSCSI. Each block device is created with a specific size and most block systems allow expansion of devices. Data is managed as blocks in tracks and sectors. Additional capacity requirements are usually met by creating and allocating new devices.

Standard file system storage allows creation of volumes and/or pools, protected with various RAID schemes, upon which file systems can be created. Data is managed as file hierarchies in folders and sub-folders. In most scenarios, file systems are mounted to hosts and accessed over IP networks using common protocols such as NFS, CIFS, and SMB, although some file system storage solutions also support access via HTTP.

Both block and file storage solutions are primarily designed to reside in single locations and to be locally network-attached to servers. In general, application servers must 'see' the block devices and/or file systems, which requires provisioning of storage to those servers and results in a tight coupling of servers and storage. Most block and file storage systems are designed as combinations of specific storage hardware and microcode developed for that hardware, packaged as a storage array. This also results in a tight coupling of storage software and hardware.

Object storage architectures generally consist of a software layer running on top of and independent from underlying hardware. Storage services (e.g. protection, replication, distribution, etc.) in the object storage software layer are abstracted from and have few or no dependencies on the hardware layer. Object storage solutions commonly leverage commodity hardware enabling reduced IT infrastructure costs and increased scale-out capability. These characteristics tend to be common in most types of software-defined storage. In addition, object storage architectures are designed to allow API-based management of data at the object level. To create, read, update, and delete objects, applications make RESTful API calls using a URL to another application (i.e. object storage services application) with no need to understand the storage type, location, features, structure, protocols, etc. Application servers do not need to 'see' the storage, do not require provisioning, mounting, or allocation of storage, and are decoupled from the storage platform. Object storage architectures incorporate distribution of data across multiple sites, allowing creation of logical pools of storage that span multiple, geo-distributed locations. Applications can manage objects in these logical storage pools

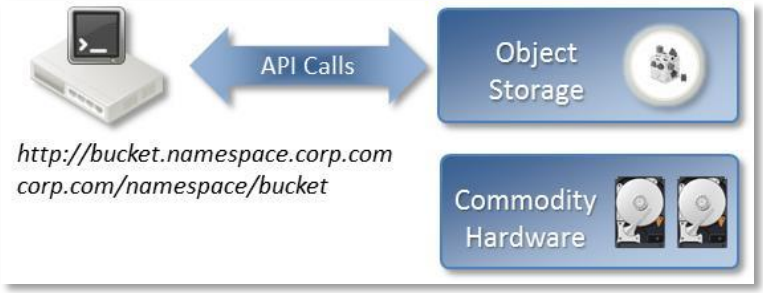


Figure 2 - Object Storage

over IP from any location and do not need to reside local to the storage system. Data protection is generally achieved in the software layer by replicating full copies of objects across nodes, performing erasure coding, and combinations of the two.

Scalability

A key differentiator between object storage and block and file storage is scalability. Traditional block storage systems are designed to scale up by adding more resources (e.g. disks, adapters, cache, etc.) to an array but not to scale out. Scaling up is limited by the specific block array capabilities, and more arrays must be implemented to support additional requirements. Newer software-defined block storage solutions, such as EMC ScaleIO®, are designed to leverage commodity hardware and support scaling out by adding more nodes. Regardless of the type of block storage system, scalability is limited by factors such as maximum number of LUNs, nodes, disks, and allocated volumes. Modern file storage solutions support both scaling up and out; however, scalability is limited by factors such as file size and maximum number of files, directories, and folders.

Object storage is designed for unlimited scale-out capability. The ability to extend an object store across multiple, globally-dispersed sites significantly reduces or eliminates scalability limitations due to location-related issues and adds data protection options. Being able to expand the capacity of an object store by adding nodes in one or more sites allows scaling out to exabyte levels. The flat nature of an object address space and the ability to access objects using unique identifiers allows for unlimited quantities of objects per object store and eliminates limitations related to hierarchical storage structures.

Performance

The difference in performance between object storage and block and file storage systems is another key differentiator. In general, block and file storage systems tend to provide higher performance than object storage solutions, although object storage vendors are working to increase performance of their systems.

Block storage systems tend to provide the highest levels of performance. Factors such as direct access to blocks of data, lack of storage-side metadata, and locality of storage to application server contribute to higher performance. In contrast, the further block storage is moved away from an application, the more the increased latency will cause performance to suffer.

For file sharing and access requirements, network-attached storage (NAS) systems such as EMC Isilon® are good options from a performance perspective. As with block systems, locality of NAS to applications and users who manage the files is key to providing high performance.

Object storage generally provides performance levels lower than that of block and file storage due to several factors. For example, regional and/or global dispersion of object data can result in higher latencies. Network issues that are not under the control of object users and applications – such as bandwidth, throughput, and contention – can negatively affect performance levels as well. Whereas performance is generally measured for block storage in single-digit milliseconds and file storage in low double-digit milliseconds, performance for object storage is generally measured in double-digit milliseconds and greater.

Use Cases

One of the primary factors in determining what type of storage to use – object, file, or block – is the use case.

Primary use cases for block storage include:

- High-performance databases
- Transactional workloads
- Mission critical applications such as Microsoft Exchange and Oracle
- Any application or system architecture that requires block devices mounted to servers
- Block-based access

Primary use cases for file storage include:

- Data lakes
- Hadoop big data analytics
- File shares and home directories
- High file performance requirements
- File-based access (CIFS, SMB, NFS)

Primary use cases for object storage include:

- Unstructured data repository
- Cloud storage platform
- Archive and tape replacement
- Mobile- and web-scale applications
- Big Data
- IoT (Internet Of Things)
- Global requirements for data stores, namespaces, replication, protection, etc.
- RESTful API-based access

In general, the best use cases for object storage center around collection and storage of massive quantities of data with simple access and scalable performance, whereas block and file storage are better for high transactional and interactive applications and use cases.

Chargeback

Chargeback models for object, file, and block storage differ based on differences in architectures and usage formulas. Models for block and file storage tend to be based on similar factors and differ significantly from object storage models.

Block storage chargeback models are generally based on presented and/or used capacity. Presented storage is the amount of block capacity visible to a server. Presented capacity consists of the capacity of one or more block devices or LUNs configured on one or more physical and/or virtual storage arrays and allocated to a server. Used capacity is the amount of capacity used by data that has been written to one or more block devices and stored in one or more array storage pools. Chargeback pricing varies based on storage-related factors such as tiers, RAID protection, raw capacity utilization, copy services, and performance. Block chargeback systems generally use application and/or server names to logically map to and determine the department, line of business, customer, etc. who should receive and is responsible for paying the bill.

File storage chargeback models are generally based on the size of file systems and/or used file system capacity. Similar to block storage chargeback models, pricing for file storage varies based on storage-related factors such as tiers, RAID protection, raw

capacity utilization, copy services, and performance. File chargeback systems generally use file system names and/or underlying volume names to logically map to and determine the department, line of business, customer, etc. who should receive and is responsible for paying the bill.

Chargeback models for object storage are generally based on capacity used, number of operations, and amount of data transferred. Pricing for object storage varies based on factors such as availability, durability, latency, and data storage location. Object chargeback systems generally use namespaces and/or buckets to logically map to and determine the department, line of business, customer, etc. who should receive and is responsible for paying the bill.

Many different types of public and private object storage solutions are available with varying features. EMC's current object storage solution is Elastic Cloud Storage, a third-generation platform designed for next-generation applications and incorporating strengths and features from its predecessor platforms – Centera® and Atmos®.

EMC Elastic Cloud Storage

EMC Elastic Cloud Storage (ECS) is a software-defined, cloud-scale, object storage platform that combines the cost advantages of commodity infrastructure with the reliability, availability and serviceability of traditional arrays."[3]

ECS allows any organization to deliver public cloud services with the control and reliability of a private cloud infrastructure. ECS provides support for multiple types of unstructured data (object, file, HDFS) and workloads on a single platform. Data can be accessed through a single namespace regardless of location. The software-defined architecture supports unlimited scalability with no single points of failure. In general, ECS allows organizations to experience cloud-scale economics in their own data centers.

Understanding ECS architecture, logical constructs, chargeback scenarios, and features related to chargeback is an important prerequisite to defining a chargeback model for ECS object storage.

---

[3] EMC. "EMC Elastic Cloud Storage (ECS) Overview and Architecture." January, 2016. Web. February 15, 2016. (EMC 2016)

Note: The use of this icon in this section represents discussions of ECS topics that relate specifically to and may be incorporated into definition of object storage chargeback models.

ECS Architecture

This section provides foundational knowledge of the ECS architecture as necessary to develop a chargeback model for ECS object storage. For a more extensive discussion of ECS architecture, please refer to the EMC Elastic Cloud Storage Overview And Architecture document.[4]

ECS is available in two forms – appliances, and software only. Appliances consist of ECS Software and EMC-supplied and supported commodity hardware. The software-only option consists of ECS Software, with the commodity hardware supplied and supported by customers. The architecture discussion in this section applies to both forms.

ECS allows management of objects through several access methods – Object (Amazon S3, EMC Atmos, OpenStack Swift, EMC Centera CAS), Hadoop Distributed File System (HDFS), and NFSv3. Object access is supported through the use of RESTful API methods such as GET, PUT, POST, LIST, etc. These access methods can be used to create, retrieve, update, and delete data, user-defined metadata, and system metadata in ECS.

All operations in ECS are performed on chunks of data. When data is stored in ECS, ECS always writes the data in 128 MB chunks. Data protection is provided on all chunks using a combination of full copies, erasure coding, and 'exclusive or' (XOR). (For a technical discussion of XOR, refer to the computer science definition at https://en.wikipedia.org/wiki/Exclusive_or.) When an application sends data to ECS, ECS writes the data into one or more chunks, triple-mirrors the chunks across three local nodes, and then sends acknowledgement of a successful write back to the application. This triple mirroring of chunks provides immediate protection in case of node and/or disk hardware failures. Following a successful write operation, ECS then executes erasure

[4] EMC. "EMC Elastic Cloud Storage (ECS) Overview and Architecture." January, 2016. Web. February 15, 2016.

coding on the chunks and deletes the three full copies. The standard ECS erasure coding is based on the Reed Solomon 12+4 erasure-coding scheme which breaks chunks into 12 data fragments and four coding (or, parity) fragments. The 16 fragments are distributed across the nodes in the local cluster, providing protection in case of node and/or disk hardware failures with greater efficiency than triple mirroring. In a scenario where ECS is installed in one site (or installed in multiple sites but replication is configured for single-site protection), the standard erasure coding results in a 1:1.33 ratio of data to capacity required for storing and protecting the data.

ECS also provides a cold archive feature, in which the erasure coding breaks chunks into 10 data fragments and two coding fragments. Cold archive erasure coding is targeted at specific use cases that can tolerate more efficient storage of data with less protection. In a scenario where ECS is installed in one site (or installed in multiple sites but replication is configured for single-site protection), cold archive erasure coding results in a 1:1.2 ratio of data to capacity required for storing and protecting the data.

In an architecture where ECS is installed in two sites and replication is configured to span both sites, the erasure coding occurs on the site where the object is initially written. Following successful erasure coding, the erasure-coded fragments are copied to the other site providing protection for a single-site failure. In this scenario, the standard erasure coding and mirroring of fragments across the two sites results in a 1:2.67 ratio of data to capacity required for storing and protecting the data. When using cold archive erasure coding, the ratio is 1:2.4.

In an architecture where ECS is installed in three or more sites and replication is configured to span at least three sites, ECS uses an ‘exclusive or’ (XOR) operation to achieve greater storage efficiency without sacrificing protection. As with the previous scenarios, erasure coding occurs on the site where the object is initially written. Following successful erasure coding, a secondary copy of each chunk is made on one of the other sites. When enough chunks of different objects are available on a site, ECS will automatically perform a XOR on those chunks, place the result on that site, and remove the original chunks. The more sites across which chunks are replicated, the greater the resulting storage efficiency (as shown in *Table 1 - Storage Efficiency Per Number Of Sites In A Replication Group*).

Number of Sites in a Replication Group	Standard Erasure Coding Storage Efficiency	Cold Archive Erasure Coding Storage Efficiency
1	1.33	1.20
2	2.67	2.40
3	2.00	1.80
4	1.77	1.60
5	1.67	1.50
6	1.60	1.44
7	1.55	1.40
8	1.52	1.37

Table 1 - Storage Efficiency Per Number Of Sites In A Replication Group[5]

As an option with architectures consisting of three or more sites, ECS can be configured to store a full readable copy of each object in each site in a Replication Group (refer to the *Replication Group* section of this document for a description). When enabled, this option provides better read performance in secondary sites at the cost of higher storage requirements. Note: When enabled, this option disables the XOR operation for that Replication Group.

The number of sites in an ECS object store and the configuration of Replication Groups directly affects the amount of capacity used to store and protect objects, which in turn may affect the total fee that the object owner is charged for storing and managing those objects.

ECS Concepts and Logical Constructs

Each ECS deployment is designed and configured using several logical building blocks, most of which affect and should be incorporated into any object storage chargeback model.

Virtual Data Center

As stated previously, ECS is available in two forms – appliances and software only. Appliances consist of ECS Software and EMC-supplied and supported commodity hardware, while the software option consists of ECS Software only. Regardless of the

[5] EMC. "EMC Elastic Cloud Storage (ECS) Overview and Architecture." January, 2016. Web. February 15, 2016.

form, each location where an instance of ECS is installed is generally referred to as a site. In a single site, the collection of appliances or customer-supplied commodity hardware networked together and managed by ECS software is known as a Virtual Data Center (VDC).

VDCs can be implemented in multiple sites and logically linked together, creating a geo-federation. Up to eight globally-dispersed VDCs can be included in a geo-federation.

Storage Pool

A storage pool is a logical grouping of commodity hardware nodes within a VDC. At least one storage pool must be defined per VDC and at least four nodes must be included in each storage pool. Configuration of multiple storage pools within a VDC allows the physical separation of object data across different sets of hardware, as is sometimes required in multi-tenant environments or by certain applications.

Two erasure coding options are available and configured at the storage pool level – standard and cold archive, both of which are defined previously in this document in section *ECS Architecture*. Standard erasure coding requires storage pools to be configured with at least four nodes, and cold archive erasure coding requires storage pools to be configured with at least eight nodes.

Understanding ECS storage pools is important when developing a chargeback model. Specifically, erasure coding options can be used to offer different levels of storage efficiency and protection which may in turn result in different cost amounts that must be incorporated into the chargeback model.

Replication Group

A replication group is a logical grouping of one or more storage pools. A replication group can consist of storage pools:

- Locally, within the same VDC – protects against disk and node failures
- Globally, across multiple VDCs – protects against disk, node, and site failures

The purpose of replication groups is to define the storage pools where object data is stored, protected, and available to be accessed by applications. At least one replication group must be defined, and multiple replication groups may be configured to address multiple sets of requirements including physical separation of data in multi-tenant

environments, storage of objects in specific locations, protection and replication options, and more.

Understanding ECS replication groups is important when developing a chargeback model. Specifically, replication groups control chargeback-related factors including:

- Protection – prevent data loss from disk, node, and/or site failures
- Replication – number of VDCs across which object data is replicated
- Local copies – whether complete, readable copies of objects are stored in each storage pool in a replication group

Each of these configuration options may be used when developing a chargeback model. For example, you may want to offer different protection levels at different costs. The number of sites across which object data is replicated may be used as a basis for offering multiple service levels, each at a different cost. Enabling the option to store complete, readable copies of objects in each storage pool location may optimize read performance but may also result in higher storage utilization and costs.

Namespace

A namespace is a logical collection of names. In object storage architectures, a namespace provides a view into a collection of objects. Object storage solutions are designed to provide global namespaces, which provide single consolidated views of objects stored across multiple locations. At least one namespace must be defined when deploying ECS, and multiple namespaces may be configured to address multiple sets of requirements. Each namespace consists of one or more replication groups.

In multi-tenant ECS environments, namespaces can represent and be owned by enterprises, business units, departments, or any grouping of users. A common model for creating namespaces is for each namespace to represent a tenant, although other models are possible. Users and permissions are defined within namespaces, allowing users to manage objects within each namespace. Users defined in one namespace cannot access objects in any other namespace.

Understanding ECS namespaces is important when developing a chargeback model. In general, two options exist for how namespaces can be defined – one

namespace per tenant, or namespaces that are shared by multiple tenants. If one namespace is defined per tenant, a chargeback model can be defined to evaluate each namespace to determine which tenant to charge back for storage utilization. If multiple tenants share a namespace, other constructs or metadata (for example, bucket name and/or bucket owner) must be evaluated to determine whom to charge back for storage utilization.

Bucket

A bucket is a logical container within a namespace in which objects may be stored. As with namespaces, users with permissions to access objects within buckets in a namespace cannot access buckets or objects in any other namespace. Permissions can be defined to grant namespace users permissions to create/delete buckets, manage permissions for buckets, and access objects in buckets. In some use cases, applications automatically create, manage, and delete buckets as needed. Another common use case for creating buckets is for each bucket to be created for and represent a sub-tenant. Many other bucket use cases are possible as well and can be used in combination depending on requirements.

Understanding ECS buckets is important when developing a chargeback model. In a tenant scenario where multiple tenants share a namespace, one method to determine who should be charged for object store utilization is to evaluate bucket names and map them back to the tenant, business unit, consumer, etc. who owns and utilizes each bucket. If bucket names cannot be used to determine whom to charge back (for example, bucket naming standards or metadata is not available), bucket owners must be evaluated and mapped back to the tenant, business unit, consumer, etc. who owns and utilizes each bucket.

Users

Three types of users must be defined in ECS – object users, namespace administrators, and system administrators.

Object users are the users of object data stored in ECS and can be given permissions to read and write objects as well as create and delete buckets within a namespace. Reading and writing of objects by object users is performed by using one or more of the access methods supported by ECS, including Amazon S3, OpenStack Swift, EMC CAS,

EMC Atmos, HDFS, and NFS. Object users who create buckets become the owner of the buckets they create.

Namespace administrators manage settings and object users for a specific namespace, while system administrators configure and manage the ECS system and can also function as namespace administrators.

Understanding ECS users is important when developing a chargeback model. In a tenant scenario where multiple tenants share a single namespace, one of two ways to determine who should be charged for object store utilization is to evaluate object users who are bucket owners and map them back to the tenant, business unit, consumer, etc. who owns and utilizes each bucket (the other way is to evaluate bucket names).

Multi-Tenancy and Tenant Scenarios

ECS supports multiple tenant scenarios for multi-tenant environments. Four common tenant scenarios[6] are:

- Enterprise single tenant
- Enterprise multiple tenant
- Cloud Service Provider single tenant
- Cloud Service Provider multiple tenant

These scenarios are described in *Table 2 - Tenant Scenarios*.

[6] EMC. "Elastic Cloud Storage (ECS) Version 2.2 Administrator's Guide." December, 2015. Web. February 15, 2016. (EMC 2015)

Scenario	Number of Namespaces	Namespace Administrator	Number of Buckets	Bucket Owners	'Chargeback By' Options
Enterprise – single tenant	One (shared by all users)	IT Department	One or more per sub-tenant	Options: • IT Department • Sub-tenant	Bucket, Bucket Owner
Enterprise – multiple tenants	One per tenant	IT Department	Options: • One or more per user • One or more per sub-tenant	Options: • IT Department • User • Sub-tenant	Namespace
Cloud Service Provider – single tenant	One (shared by all users)	Service Provider	Options: • One or more per user • One or more per sub-tenant	Options: • User • Sub-tenant	Bucket, Bucket Owner
Cloud Service Provider – multiple tenants	One per tenant	Tenant	Options: • One or more per user • One or more per sub-tenant	Options: • User • Sub-tenant	Namespace

Table 2 - Tenant Scenarios

Each scenario is defined by a unique combination of factors relating to tenants, namespaces, buckets, and administration. Factors addressed in *Table 2 - Tenant Scenarios* are defined as follows, along with sample questions that must be answered when determining how to design the model that will best meet your requirements:

- Number of Namespaces – the quantity of namespaces to be created
    - Will all tenants share the same namespace(s)?
    - Will each tenant be assigned their own dedicated namespace?
- Namespace Administrator – the group, team, department, etc. who will be given administrative permissions for each namespace
    - Will IT departments create and retain administrative control of all namespaces?
    - Will the object store be operated in a service provider model where administrative control of each namespace is given to the tenant for which the namespace was created?

- Number of Buckets – the quantity of buckets to be created in each namespace
    - Will one bucket represent and be created per tenant?
    - Will buckets represent and be created per sub-tenant?
    - Will buckets represent and be created per application?
    - Will individual object users be able to create buckets as needed?
- Bucket Owner – the user(s) who will be given permission to create and manage buckets
    - Will individual object users be permitted to create their own buckets as needed?
    - Will applications be assigned their own service account object user and be permitted to create buckets automatically?
    - Will permission to create buckets be given to one user, a limited set of users, or to any object user defined in the namespace?

Combinations of these factors and/or business requirements other than those listed in *Table 2 - Tenant Scenarios* may result in the need to define other tenant scenarios.

The design of the tenant scenario(s) directly affects the design of a chargeback model. As discussed in section *How Does Object Storage Compare To Block & File Storage?*, object storage is not provisioned to servers; therefore, charging back based on server name is not possible. Charging back for utilization of object storage must be based on constructs and metadata available within the object store itself, such as:

- Namespace name
- Bucket name
- Bucket owner
- Bucket tags

Depending on the specific tenant scenario, one or more of these items may be used to determine who should be charged back for utilization, as shown in *Table 2 - Tenant Scenarios*.

Chargeback Scenarios

In tenant scenarios where each namespace represents and is dedicated to a

single tenant, the chargeback model should be based on namespace name. Each tenant would be charged back for total capacity utilization within its namespace(s).

In tenant scenarios where multiple tenants share the same namespace, charging back based on namespace is not feasible and must be based on bucket-level information. In shared-namespace scenarios, buckets must represent or logically map to tenants or sub-tenants in order to support charging back for utilization. Each tenant would be charged back for total capacity utilization within its bucket(s).

In addition to bucket names and owners, ECS provides an additional feature that can be used for chargeback – bucket tagging. "Tags" in the form of name-value pairs can be assigned to a bucket using the ECS Portal or the ECS Management REST API, enabling object data stored in the bucket to be categorized. For example, bucket data can be associated with a cost-center or project. Bucket tags and values can be read and managed using the ECS Portal or using custom clients with the ECS Management REST API. In addition, bucket tags are included in the metering data reports in the ECS Portal or ECS Management REST API."[7] Bucket tags can be used to define chargeback-related information on which a chargeback model could be based.

Note: Charging back based on object-level information is not possible. Since ECS supports unlimited quantities of objects and their nature can be volatile (i.e. some applications create and delete large numbers of objects frequently), ECS does not support monitoring and reporting on individual objects.

ECS supports the creation of reports using external tools by exposing all configuration information through the management REST API. Any tool that supports execution of RESTful API commands can be used to extract data from ECS. The extracted data can then be used by any report development tool for creation of inventory, configuration, capacity, performance, and chargeback reports. However, one of the IT industry's top-rated storage resource management solution – EMC ViPR SRM – already provides dozens of built-in ECS reports as well as a powerful data collection engine and robust report customization solution.

---

[7] EMC. "Elastic Cloud Storage (ECS) Version 2.2 Administrator's Guide." December, 2015. Web. February 15, 2016.

Chargeback Reporting & ViPR SRM

EMC provides three reporting options for ECS:

- ECS Portal
- ECS Monitoring & Reporting
- ViPR SRM

The ECS Portal is included with ECS and provides monitoring, diagnostics, and event auditing. Monitoring pages provide overviews of storage, resources, services, and events. The Monitoring pages allow you to drill down to get the desired view of diagnostic data. The Dashboard is the first page you see upon logging into the portal. It provides a quick summary of monitoring data.

Figure 3 - ECS Portal

ECS Monitoring & Reporting (M&R) is included with ECS and provides additional reporting features as compared to the ECS Portal. ECS M&R requires installation on virtual or physical hosts external to ECS and provides reporting on:

- Storage capacity utilization
- Object creation and deletion
- Hardware, network, disk, node, and performance health
- Statistics per node, including process health, traffic metrics, and resource utilization
- Replication statistics
- Erasure coding
- Single view of multiple VDCs

Refer to *Figure 4 - ECS Monitoring & Reporting* for a sample view.

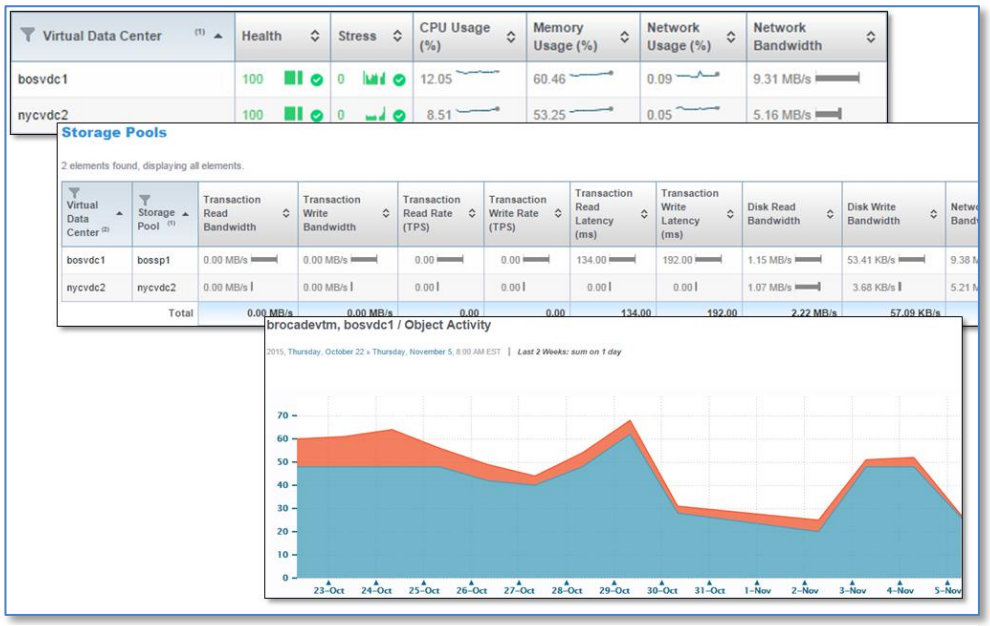


Figure 4 - ECS Monitoring & Reporting

ViPR SRM is an enterprise-class comprehensive monitoring and reporting solution, and provides the most extensive reporting capabilities for storage infrastructures, including ECS. ViPR SRM is an optional licensable software solution that requires installation on virtual or physical hosts external to ECS. In addition to all the ECS Monitoring & Reporting features and reports, ViPR SRM includes additional enterprise-class capabilities. Enterprise-wide views into block, file, and object storage infrastructures are supported, as well as built-in chargeback reports (refer to *Figure 5 - ViPR SRM Object Dashboard* for a sample view). The rest of this section will focus on using ViPR SRM to produce chargeback reports for ECS object storage.

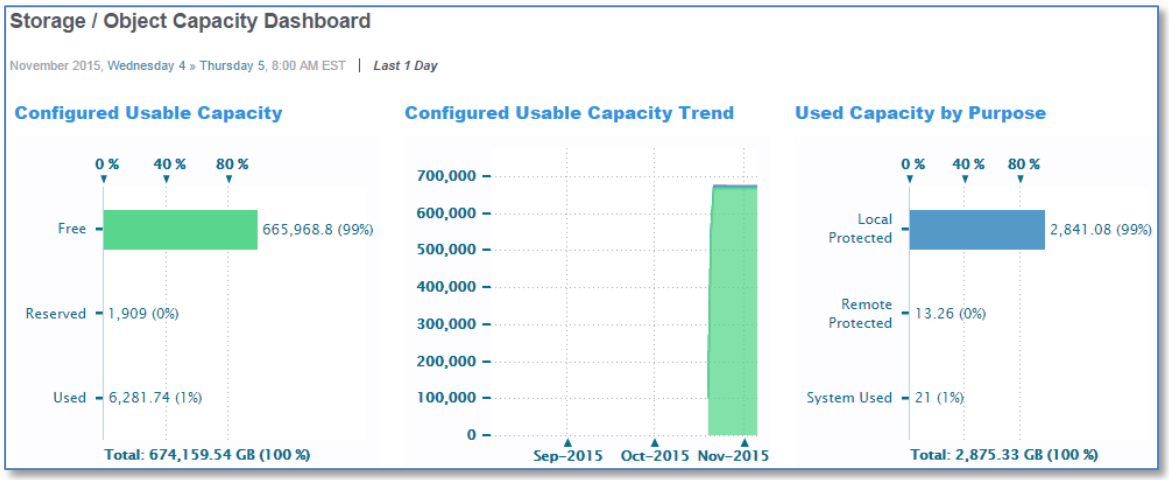


Figure 5 - ViPR SRM Object Dashboard

Whether you can use any of the built-in object storage chargeback reports in ViPR SRM or will need to create custom chargeback reports depends on the chargeback model developed to meet your enterprise's business and technical requirements.

Chargeback Model Examples

Chargeback models implemented by public and private cloud vendors and enterprises span the range of complexity from simple (e.g. based solely on capacity used) to highly complex (e.g. based on capacity, tiers, network bandwidth, performance, etc.). To facilitate defining of a chargeback model, this section provides various examples of public and private cloud storage chargeback models.

Amazon's public cloud object storage solution is called Simple Storage Service (S3).[8] Amazon charges for S3 usage based on the following metrics:

- Capacity used per month
- Amount of data transferred in and out per month
- Number of PUT/COPY/POST/LIST requests per month
- Number of GET and all other requests per month[9]

Amazon's tenant scenario is similar to the Cloud Storage Provider – Single Tenant scenario defined in *Table 2 - Tenant Scenarios*. All S3 users share the same namespace and are charged back by capacity utilization at the bucket level. Specific pricing and chargeback models differ for each of three different tiers, or classes, of S3 storage.

Google's public cloud object storage solution is called Google Cloud Storage (GCS).[10] Google charges for GCS usage based on the following metrics:

- Capacity used per month
- Amount of data retrieved per month

---

[8] "Amazon S3" Amazon, 2016. Web. February 15, 2016. (Amazon 2016)

[9] "Amazon S3 Pricing" Amazon, 2016. Web. February 15, 2016. (Amazon 2016)

[10] "Google Cloud Storage" Google, 2016. Web. February 15, 2016. (Google 2016)

- Number of Class A operations per month (e.g. GET service, GET bucket, PUT, POST)
- Number of Class B operations per month (e.g. GET object, HEAD)[11]

Google's tenant scenario is similar to the Cloud Storage Provider – Single Tenant scenario defined in *Table 2 - Tenant Scenarios*. All GCS users share the same namespace and are charged back by capacity utilization at the bucket level. Specific pricing and chargeback models differ for each of three different tiers of GCS storage.

As an example of a private cloud chargeback model currently in use by an EMC customer who deployed ECS, their IT department defined a model that consists of two tiers (single-site protection and multi-site protection) and is based on capacity used per bucket. This is a relatively simple model; however, it meets all their business and technical requirements.

Chargeback Metrics

Of the metrics commonly used by public cloud object storage providers for chargeback (refer to section *Chargeback Model Examples*), ViPR SRM supports built-in reporting on the following ECS metrics:

- Capacity used per time period
- Amount of data retrieved per time period
- Number of objects created
- Number of objects deleted

As of the writing of this article (Feb. 2016), ECS does not track the number of individual RESTful API methods (such as GET, PUT, POST, LIST, etc.) executed so these metrics are not available in ViPR SRM.

Multiple tiers of object storage can be defined in ViPR SRM using object service levels, and chargeback models can be based on namespaces and/or buckets depending on the specific ECS architecture design.

---

[11] "Google Cloud Storage Pricing" Google, 2016. Web. February 15, 2016. (Google 2016)

Chargeback Model Parameters

When developing a chargeback model, multiple parameters must be considered and chosen for incorporation into the formulas that make up the model. Common parameters include:

- Unit of measurement
- Time period
- Aggregation – average, min, max
- Storage service level or class

The most common unit of measurement incorporated into object storage chargeback models is gigabyte (GB), where 1 GB equals 1024^3 bytes (this unit of measurement is more accurately known as a gibibyte, or GiB). Other units of measurement may be used, such as terabyte (TB) or megabyte (MB), but those units tend to be either too large resulting in lack of granularity or too small resulting in unwieldy numbers.

The most common time period used in object storage chargeback models is a calendar month. This time period is common across billing for many types of products and works well when billing for object storage utilization.

The most common type of aggregation used in object storage chargeback models is arithmetic mean, or average, based on daily utilization. Other types of aggregation may be used, such as maximum or minimum, but average is used by most public cloud providers.

Whereas these three parameters have been standardized in the industry (i.e. cost per GB per month based on average daily utilization), the parameter that differs the most across providers is storage service level, also known as class or tier. Both Amazon and Google offer three storage service levels with various differences:

- Amazon S3: Standard, Standard Infrequent Access, and Glacier[12]
- Google Cloud Storage: Standard, Durable Reduced Availability, and Nearline[13]

---

[12] "Amazon S3 Storage Classes" Amazon, 2016. Web. February 15, 2016. (Amazon 2016)

When defining service levels based on an ECS architecture, several factors should be considered:

- Protection
  - Single site – provides protection for node and disk failure
  - Multiple sites (from two to eight sites) – provides protection for node, disk, and site failure
- Replication
  - Single site – single storage pool with one readable copy
  - Multiple sites
    - Standard – one readable copy in primary object site with mirrored or XOR'd chunks in other sites for recovery
    - Full readable copies – a full readable copy of each object in each site where the storage pools in a replication group are located
- Erasure coding
  - Standard – a 12+4 scheme with 12 data fragments and four coding fragments, for a 1:1.33 ratio of data to capacity
  - Cold Archive – a 10+2 scheme with 10 data fragments and two coding fragments, for a 1:1.2 ratio of data to capacity
- Storage efficiency ratio – refer to *Table 1 - Storage Efficiency Per Number Of Sites In A Replication Group*

As an example, the following service levels could be defined in an ECS object store:

- Enhanced
  - Protection – three sites for disk, node, and site failure protection
  - Erasure coding – standard 12+4
  - Replication – full readable copies in all three sites
- Standard
  - Protection – three sites for disk, node, and site failure protection
  - Erasure coding – standard 12+4

---

[13] "Google Cloud Storage" Google, 2016. Web. February 15, 2016.

    - Replication - one readable copy in primary object site with XOR'd chunks in other two sites
- Archive
    - Protection – one site for disk and node failure protection
    - Erasure coding – standard 12+4
    - Replication – one readable copy in primary object site

Many combinations of ECS features, architecture designs, and other factors may be used to define a menu of object storage services, and the capability of producing chargeback reports is critical to operationalizing object storage as a service.

ViPR SRM Built-In Object Storage Chargeback Reports

ViPR SRM provides many built-in object storage reports, including several chargeback reports. In addition to the built-in reports designed to address approximately 80% of common IT reporting requirements, ViPR SRM allows custom data collections and reports to be developed to meet additional requirements. This section will focus on using and customizing the built-in object chargeback reports.

The following built-in object storage chargeback reports are provided with the ECS SolutionPack for ViPR SRM:

- Chargeback By Namespace
- Chargeback By Bucket
- Chargeback By Service Level

Each of these reports are designed to meet the majority of object storage chargeback reporting requirements for one or more of the tenant scenarios described in *Table 2 - Tenant Scenarios*.

The Chargeback By Namespace report is designed to meet chargeback requirements for the following two tenant scenarios:

- Enterprise – multiple tenants
- Cloud Service Provider – multiple tenants

These two tenant scenarios reflect ECS architectures that are configured with multiple namespaces – one per tenant. The Chargeback By Namespace report provides details

per namespace with various metrics for which tenants can be charged, as shown in the sample report in *Figure 6 - Chargeback By Namespace*.

Namespace (1)	Local Protected	Remote Protected	Other Service Levels	No. Objects	No. Objects Created	No. Objects Deleted	Data Downloaded	Data Uploaded	Total Cost ($)
brocadevtm	603.01 MB	0.00 GB	0.00 GB	660	620	95	28.60 KB	54.33 KB	0.03
chrisns	0.00 GB	288.00 KB	0.00 GB	2	0	0	0.00 MB	0.00 MB	0.00
cosbenchns	0.00 GB	0.00 GB	0.00 GB	0	0	0	0.00 MB	0.00 MB	0.00
ecsgw	0.00 GB	0.00 GB	0.00 GB	1	1	0	0.00 MB	0.00 MB	0.00
isiloncloudpool	314.39 MB	0.00 GB	0.00 GB	720	1,640	0	174.09 MB	696.35 MB	0.02
namespace1	83.81 GB	0.00 GB	0.00 GB	104,913	502	86	9.06 MB	4.76 MB	4.19
sales	6.70 GB	10.68 GB	0.00 GB	50	4	0	0.00 MB	6.78 GB	1.40
Total	91.41 GB	10.68 GB	0.00 GB	106,346	2,767	181	183.17 MB	7.46 GB	5.64

Figure 6 - Chargeback By Namespace

The metrics in this report include:

- Total size of objects (GB) in a namespace by service level (i.e. Local Protected, Remote Protected, and Other Service Levels)[14]
- Total object count in the namespace
- Number of objects created and deleted during the reporting period
- Data uploaded and downloaded during the reporting period
- Computed chargeback costs[15]

This report supports drill-down into each namespace to see bucket and service level chargeback information for that namespace, including the Chargeback By Bucket report as shown in *Figure 7 - Chargeback By Bucket*.

[14] EMC. "EMC ViPR SRM Version 3.7.0.0 Chargeback Guide." October, 2015. Web. February 15, 2016. (EMC 2015)

[15] EMC. "EMC ViPR SRM Version 3.7.0.0 Chargeback Guide." October, 2015. Web. February 15, 2016.

Bucket (2)	Virtual Data Center	Owner (1)	Replication Group	Service Level	Used	No. Objects	No. Objects Created	No. Objects Deleted	Data Downloaded	Data Uploaded	Total Cost ($)
sal-bucket1	bosvdc1	sal-user2@escala.com	georg	Remote Protected	6.48 GB	11	0	0	0.00 MB	0.00 MB	0.65
sal-bucket3	bosvdc1	sal-user2@escala.com	bosrg	Local Protected	6.01 GB	4	0	0	0.00 MB	0.00 MB	0.30
sal-bucket4	bosvdc1	sal-user2@escala.com	georg	Remote Protected	2.00 KB	2	0	0	0.00 MB	0.00 MB	0.00
sal-bucket5	bosvdc1	sal-user2@escala.com	georg	Remote Protected	1.08 GB	2	0	0	0.00 MB	0.00 MB	0.11
sal-bucket6	bosvdc1	sal-user2@escala.com	georg	Remote Protected	1.08 GB	1	0	0	0.00 MB	0.00 MB	0.11
				Total	14.65 GB	20	0	0	0.00 MB	0.00 MB	1.16

Figure 7 - Chargeback By Bucket

The Chargeback By Bucket report includes the same metrics as the Chargeback By Namespace report and adds the Bucket Owner column; however, the metrics are broken out by bucket within the specific parent namespace.

The Chargeback By Service Level report lists each service level in a specific namespace, the amount of used capacity per service level, and the cost for that used capacity (refer to *Figure 8 - Chargeback By Service Level*).

Service Level	Used Capacity	Cost per GB ($)	Total Cost ($)
Local Protected	83.81 GB	0.05	4.19
Remote Protected	0.00 GB	0.10	0.00
		Total	4.19

Figure 8 - Chargeback By Service Level

If your ECS architecture is designed to address one of the following tenant scenarios (listed in *Table 2 - Tenant Scenarios),* your ECS architecture may have one namespace that every user and application shares for bucket creation and storage of data:

- Enterprise – single tenant
- Cloud Service Provider – single tenant scenarios

The Chargeback By Bucket report for the single parent namespace would be sufficient to meet chargeback requirements for either of these scenarios.

The built-in object storage chargeback reports will meet the majority of chargeback reporting requirements; however, some situations may require custom reports.

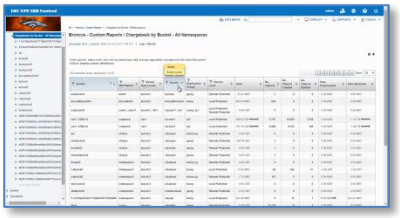
Custom Object Storage Reports

Some ECS architectures are a hybrid of the single-tenant and multiple-tenant scenarios. Some object storage requirements may drive the need for an ECS architecture with a small number of namespaces shared by multiple tenants. For example, you may need to configure one namespace per Replication Group so that each namespace reflects a specific data protection and replication configuration. In a hybrid scenario, one namespace could be configured with a default Replication Group for single-site replication (providing protection for disk and node failure) and a second namespace could be configured with a default Replication Group for three-site replication (providing protection for disk, node, and site failure). End users would be able to choose the namespace that meets their protection and replication requirements. The result in this example would be multiple tenants sharing each of these two namespaces depending on their requirements. From a chargeback perspective, an optimal report would provide a listing of all buckets and bucket owners across all namespaces and include the same metrics as in the built-in Chargeback By Bucket report. A sample Chargeback By Bucket Owner report incorporating these requirements is shown in *Figure 9 - Chargeback By Bucket Owner Report*.

Bucket (2)	Namespace	Virtual Data Center	Owner (1)	Replication Group	Service Level	Used	No. Objects	No. Objects Created	No. Objects Deleted	Data Downloaded	Data Uploaded	Total Cost ($)
bosbucket1	bosns	bosvdc1	bosuser	georg	Remote Protected	25.83 MB	5	0	0	0.00 MB	0.00 MB	0.00
brocadebuckete1	brocadevtm	bosvdc1	brocadetvmuser	bosrg	Local Protected	603.00 MB	655	0	0	0.00 MB	0.00 MB	0.03
casbucket1	casns, casns1	bosvdc1, vdc1	casuser1, root	casrg, rg1	Local Protected, Remote Protected	0.00 GB	0	0	0	0.00 MB	0.00 MB	0.00
vdc1-128M-31	cosbench	vdc1	cb-user1	rg1	Local Protected	577.13 GB	4,841	10,000	1,639	0.00 MB	1.16 TB	28.86
vdc1-128M-41	cosbench	vdc1	cb-user1	rg1	Local Protected	169.52 GB	1,422	2,892	48	0.00 MB	344.51 GB	8.48
acl	chrisns	bosvdc1	chrisuser	bosnycrg	Remote Protected	0.00 GB	0	0	0	0.00 MB	0.00 MB	0.00
chrisbucket	chrisns	bosvdc1	chrisuser	georg	Remote Protected	288.00 KB	1	0	0	0.00 MB	0.00 MB	0.00

Figure 9 - Chargeback By Bucket Owner Report

This report is custom and was developed using the built-in Chargeback By Bucket report as a baseline template with customizations made via the ViPR SRM user interface. As a supplement to this article, a video is available that shows exactly how this report was created and customized. To view the video, double-click on the video image.



As demonstrated in the video, any report can be scheduled to run on a periodic basis (e.g. monthly) and configured to email the report data in a specific format (e.g. CSV file for further manipulation in Excel by a Finance team).

Conclusion

Enterprises and service providers are increasingly deploying modern web-scale applications to increase competitive advantages, looking to save money with new archive platforms, implement object storage as a service, introduce big data analytics, and leverage the 'Internet of Things' for new revenue opportunities. These types of applications and services require fundamentally different data management architectures designed to support new development frameworks, unstructured data, massive scale, and globally accessible and distributed data, all at lower acquisition and operational costs.

Object storage is the current storage platform of choice for most next-generation applications and many public and private object storage solutions are available. ECS is EMC's third-generation object storage platform and is designed to meet the requirements of both traditional and modern applications.

Deploying ECS in your infrastructure introduces new operational challenges that must be addressed, a primary challenge being chargeback. To implement an object storage chargeback model, you need to understand the differences in technology and architecture between object storage and traditional storage platforms like block and file. You also need to understand ECS architecture, concepts, and logical constructs before you can develop a chargeback model that meets your business and technical requirements.

Finally, an IT monitoring and reporting tool such as EMC ViPR SRM will facilitate the task of chargeback reporting by providing built-in object storage reports designed to

meet the majority of reporting requirements, including chargeback, and powerful customization features that allow you to meet any unique or non-standard requirements.

Ideally, the information presented in this article and videos will enable you to easily design, implement, and operationalize an object storage chargeback model using EMC ECS and ViPR SRM.

Bibliography

Amazon. *Amazon S3.* 2016. https://aws.amazon.com/s3 (accessed February 15, 2016).

—. *Amazon S3 Pricing.* 2016. https://aws.amazon.com/s3/pricing/ (accessed February 15, 2016).

—. *Amazon S3 Storage Classes.* 2016. https://aws.amazon.com/s3/storage-classes/ (accessed February 15, 2016).

EMC. *Elastic Cloud Storage (ECS) Version 2.2 Administrator's Guide.* December 2015.

—. *EMC.* October 2015. (accessed February 15, 2016).

EMC. *EMC Elastic Cloud Storage (ECS) Overview and Architecture.* January 2016.

EMC. *EMC Reveals New Cloud Services Business.* October 2015.

Facebook. *Needle in a haystack: efficient storage of billions of photos.* 2009.

Google. *Google Cloud Storage.* 2016. https://cloud.google.com/storage/ (accessed February 15, 2016).

—. *Google Cloud Storage Pricing.* 2016. https://cloud.google.com/storage/pricing (accessed February 15, 2016).