White Paper

EMC Elastic Cloud Storage — Blueprint for Exascale Storage

Sponsored by: EMC

Ashish Nadkarni
July 2016

IDC OPINION

As businesses enter the IDC 3rd Platform era and embrace next-generation applications (NGAs – also known as cloud-native applications), they are forced to reexamine the infrastructure layer supporting such applications. NGAs are disruptive to the industry because of their design goals, which include data layer processes that do not assume infrastructure resilience – and therefore are written to interface with the storage layer via RESTful APIs. NGAs treat the infrastructure layer very differently than do current-generation applications (CGAs – aka traditional applications). To support NGAs in their full glory, storage suppliers have to take a clean-slate approach to developing next-generation "exascale" storage systems. They have to discard the limiting features of current-generation storage systems that are limited in scale and replace them with design principles that are more or less aligned with NGAs. These include:

- Horizontally and vertically scalable functionality that is implemented as a service across all nodes
- Equitable and independent handling of capacity, file size, and performance capabilities
- All data treated as equal, regardless of how it is ingested – no silos for different data types
- Designed for multi-datacenter operations – in terms of data ingest, distribution, and resiliency

As illustrated in this white paper, EMC's Elastic Cloud Storage (ECS) platform is built on these four pivotal design principles – borrowed from principles implemented by hyperscalers in their cloud storage implementation – and could be considered to be built specifically with NGAs in mind. As businesses embrace NGAs, it behooves them to evaluate object-based storage (OBS) platforms like the ECS as part of their next-generation infrastructure.

SITUATION OVERVIEW

There is a quiet transformation going on in enterprise IT today. Businesses are aggressively embracing IDC's 3rd Platform applications (referred to as next-generation applications). NGAs have the potential to reduce deployment and operational costs while propelling businesses to new levels of competitive differentiation and profitability. Embracing NGAs is therefore a given, and it is not a question of if but a question of when and how.

As NGAs gain adoption, they are slowly replacing current-generation applications – the mainstay of most enterprise IT environments. IDC believes that in the longer term, NGAs will take over the entire application landscape, compelling IT to completely overhaul the manner in which it deploys and services the supporting infrastructure.

NGAs are developed using design principles that include statelessness; sharded, geodispersed data architectures; horizontal scalability; and built-in analytics for a user-first experience. From a compute perspective, such applications are designed to run on industry-standard computing platforms without any prejudice or bias toward certain silicon architectures. NGAs are designed with data layer processes that do not assume infrastructure resilience — and therefore are written to interface with the storage layer via RESTful APIs.

NGAs have the potential to be disruptive to the infrastructure layer both operationally and organizationally:

- Operationally, NGAs change buying patterns — the supplier-buyer relationship is altered from a procurement and ongoing support perspective. It introduces a new set of supplier relationships and alters service-level agreements.
- Organizationally, NGAs force organizational boundaries to collapse even further. A software-defined infrastructure deployed on industry-standard hardware further pushes the adoption of a flat organizational structure — one in which all infrastructure functions are performed by a single organization and one in which application administrators have more control of resource provisioning and management.

Limitations of Current OBS Platforms

Most shared nothing OBS platforms in the market are certainly a step above older shared storage platforms but have limitations in terms of scalability when it comes to capacity and performance beyond a certain level. For example:

- Their performance starts to taper off as the number of objects surpasses billions or capacity surpasses a few petabytes.
- They can be optimized to ingest either large files or small files but not to handle both concurrently.
- They cannot handle multiprotocol ingest natively, resulting in an overhead when such ingest is employed.
- There is high potential for a system imbalance where consistent sharding can result in certain nodes getting "hot" or "overloaded."

Such limitations stem from the fact that many such systems are based on a master-slave design, dedicated name nodes, or centralized metadata database nodes. In such platforms, the metadata is stored in the memory of a single host (or cluster of hosts). As the number of objects increases, the name node or metadata node becomes overloaded and hence singularly contributes to the performance degradation of the entire system. Similarly, when the name node goes down, the system loses its intelligence until the name node functionality is fully restored. The simple object sharding schemes employed by such systems also force a trade-off between capacity and performance with regard to file size.

Defining "Exascale" Storage — Designed for Next-Generation Applications

Since NGAs treat the infrastructure layer very differently than CGAs, to support NGAs in their full glory, storage suppliers have to take a clean-slate approach to developing next-generation "exascale" storage systems. They have to discard the limiting features of current-generation object storage systems and embrace radical design principles that are more or less aligned with those adopted by developers of NGAs. Incorporating such design principles means that the system should:

- Make every functionality horizontally and vertically scalable. Each functionality should therefore be implemented as a data and compute service that scales across all nodes.
- Treat capacity, file size, and performance on par with each other. The variability of one should not compromise the scalability of the other two.
- Treat all data as equal, regardless of how it is ingested. In other words, there should not be any vertical silos for different data types. This allows the use of this platform for CGAs and NGAs.
- Be built from the ground up to include multi-datacenter operations – in terms of data ingest, distribution, and resiliency.

An object-based data layout (also known as a key/value-based data layout) that is designed with a truly scalable data and metadata distribution scheme is gaining popularity as a scalable data persistence scheme in IDC's 3rd Platform era. It bodes well for NGAs because such applications treat the storage system as a higher-level service that is accessible via RESTful APIs (known as object protocols), unlike CGAs that choose to access storage via known block or file protocols.

The EMC Elastic Cloud Storage Platform

Elastic Cloud Storage is EMC's third generation of object-based storage platform – multipurpose, enterprise-class smart storage built specifically to support NGAs. IDC believes that the approach taken by EMC in designing ECS makes it one of the best OBS platforms in the market. ECS is therefore well suited for globally dispersed or stateless NGAs that are often deployed in containers and access data via REST APIs. ECS is designed to scale effortlessly; provide scalable, geodispersed global access to data; and provide the economics of public cloud storage even in geodispersed configurations. Key differentiators of ECS from a customer perspective are:

- **Deployment flexibility.** Designed as a software-defined storage platform, ECS provides customers the ability to deploy this platform on their terms – as a turnkey storage appliance or as a software-only solution designed to run on industry-standard hardware. ECS was built from the ground up to run on a converged infrastructure stack using container technologies like Docker. ECS is deployed and automated using Docker as a framework for deploying software only in a distributed infrastructure.
- **Unprecedented scalability.** ECS is designed to be globally dispersed, "exabyte scale" storage that can handle billions of multisize objects. It features a flexible and layered software-defined architecture in which each layer is completely abstracted and independently scalable, with high availability and no single point of failure. Furthermore, ECS is optimized to support ingest of and access to both small and large files with strong global consistency.
- **Multiprotocol enterprise data lake platform.** ECS is built from the ground up to natively support multiple (object and file) data access protocols on a single platform. This eliminates the need for separate add-on gateways that become a bottleneck or single point of failure.

- **Secure and compliant data governance.** Unlike public cloud storage, ECS does not carry data residency and compliance risks. It further simplifies data governance and management with instant metadata search, analytics enabled by HDFS, and built-in optimizations for speed and storage efficiency. In addition, ECS comes with enterprise-grade features for protection, availability, encryption, authentication, and access controls, making it a highly secure platform.
- **Ease of management.** Like public cloud storage, a globally distributed infrastructure built using ECS is simple to manage as a singular entity from a central location.

Key Features of the ECS Architecture

ECS is designed as a strongly consistent shared nothing architecture that leverages industry-standard hardware. According to IDC, the characteristics of ECS discussed in the sections that follow are the key differentiators of this platform vis-à-vis other platforms in the market.

Scalable Indexing

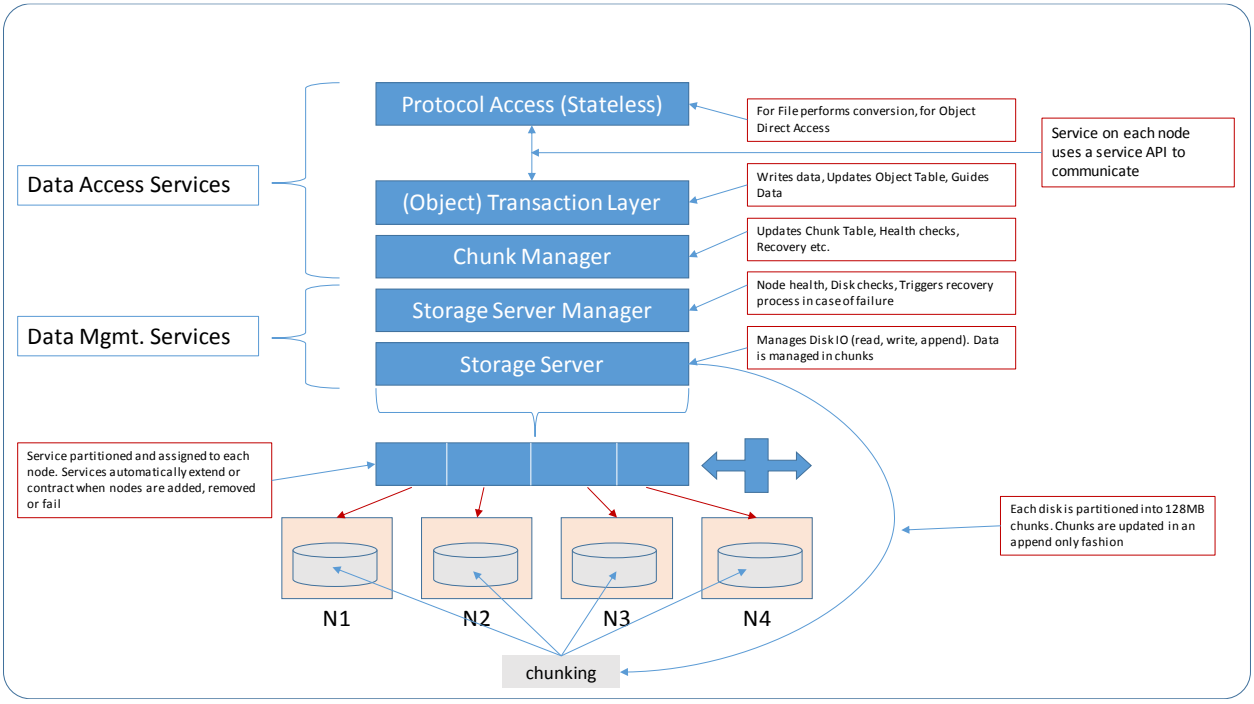
ECS utilizes a scalable indexing scheme implemented in the form of a horizontal scaling design. Figure 1 illustrates the manner in which services are implemented in ECS and how data is managed on the persistence layer (disk or SSD).

In the horizontal scaling design:

- All nodes in the cluster are treated as equal (i.e., there are no "special purpose" nodes in the cluster). All nodes participate in the cluster in an equitable fashion. Each node therefore allocates the same amount of compute, memory, and data persistence (storage) resources to run the core ECS data and metadata services. The manner in which data is distributed ensures that any number of nodes can go bad, and yet the entire system constantly moves to a good set of nodes. Over 50% of the nodes have to concurrently go bad before there is severe service degradation.
- The core ECS services – which include data and metadata services – are themselves implemented as logical key/value tables (as opposed to being implemented in the memory of a single name node). Furthermore, each table is sharded into multiple partitions (using SHA-256) and distributed onto each of the nodes. Each service communicates with the other services using a service API. The service API aggregates these "partitions" and provides uniform access independent of the nodes that service the incoming request. All services are stateless.
- Ownership of a particular service is based on availability and load. When nodes fail, the service automatically reconfigures itself, transferring ownership of the incoming transaction and, in the process, ensuring that the API access remains uninterrupted. When new nodes are added, all services automatically "extend" themselves onto to the new nodes, thereby making equitable use of their resources. This means that the system has no internode or interservice communication bottlenecks or single points of failure.

FIGURE 1

ECS Service Architecture



Source: EMC, 2016

Each key/value pair service falls into one of the two functional categories (refer back to Figure 1):

- **Data access services are a higher-layer set of services that are transactional in nature and do not concern themselves with storage resiliency.** Such services are more concerned about the scalable namespace and scalable transaction and offload the data persistence problem to the bottom layer. This category includes the protocol access service (which maps the file schematics to the object ID), object transaction service (which maps the object to the locations of the chunks), and the chunk manager (which maps the chunks to the nodes).
- **Data management services are a bottom level of services that manage the resiliency of the system, protecting the data (via replicas or erasure coding) regardless of the data type (user data, metadata, and system data).** The storage service category includes the storage server manager (which performs health checks on each of the nodes) and the storage server (which manages disk I/O).

All nodes equally share the responsibility of serving partitions (the table that contains the mapping of the partitions to the service node is maintained separately and is itself chunked across all nodes). This allows the ECS platform to maintain ACID properties. Partition tables are maintained in a manner similar to journals in a logging file system. The coming transactions will be logged to a journal chunk first and then cached in memory. Once the memory reaches some threshold or after a certain period of time, the cached transactions will be dumped to B+ tree. Both journal and B+ tree data are also persisted to chunk, thus protected by chunk against node/disk failure.

Why It Matters

Some OBS platforms in the market utilize a consistent hashing scheme, while others use master/slave or primary/secondary architectures. This presents scalability limitations because of indexing of metadata and can cause load mismatch and hot spots in the system. The key/value pair-based indexing mechanism utilized by the ECS architecture eliminates such limitations, making the platform highly scalable, with object capacities in the billions.

This design also allows ECS to be flexible from a data durability (replicas versus erasure coding) and consistency (strong versus eventual consistency) perspective. It also makes ECS provide performance parity for small and large objects. ECS automatically detects concurrent writes and aggregates them in memory and only does a single commit to the persistence layer (to the chunk).

Since there is no notion of front-end or back-end nodes, the ECS architecture can be considered to be symmetric in nature. The scheme deployed by ECS, known as indirection, makes the architecture future proof.

Data Scaling (System, Meta, and User Data)

In ECS, all system, meta, and user data is stored (read, written, or appended) in 128MB chunks — which consist of a number of blocks stored on persistent media (disk or SSD). The space (128MB) in the chunk is a logical abstraction, and the actual blocks of the chunk could be on different nodes/disks; it is not required for blocks of a chunk to be physically contiguous.

Chunks are managed by the chunk manager. Each chunk is assigned an ID, which is also known as a UUID. Object-to-chunk ID and chunk ID-to-node mappings are maintained in the two different data access tables and accessed via the respective service APIs. The writer (which is part of the storage service) sends the request to the storage server to write replicas in chunk. If any writes are incomplete, the current chunk being written is sealed and a new chunk is used in order to ensure that data written to the chunk always has enough protection.

In ECS, object metadata is stored in the same set of tables where the object-to-chunk information is stored. In a broad sense, object metadata includes properties such as timestamp of the object, which is visible to the application, and object-to-chunk mapping. Those tables are partitioned and distributed across all nodes in the cluster. This metadata is stored in the form of an index that can be accessed independent of the data (objects).

All write-to-table and write-to-chunk operations that go to different writers/tables are performed in parallel (i.e., all operations belonging to a single write and operations to different keys stored in the same table are performed in parallel).

Why It Matters

Other OBS platforms have a fixed scheme for storing metadata, which limits their scalability when the number of objects becomes large or several write operations require the databases to be updated in parallel.

In ECS, the metadata indices function like a search engine (i.e., they are highly scalable and programmatically accessible). This is of growing importance in 3rd Platform applications (NGAs) where the metadata is rich and often used in a programmatic fashion to operate on a large number of objects or to gather information on them without actually accessing each object individually.

Common Engine (Multiprotocol Access)

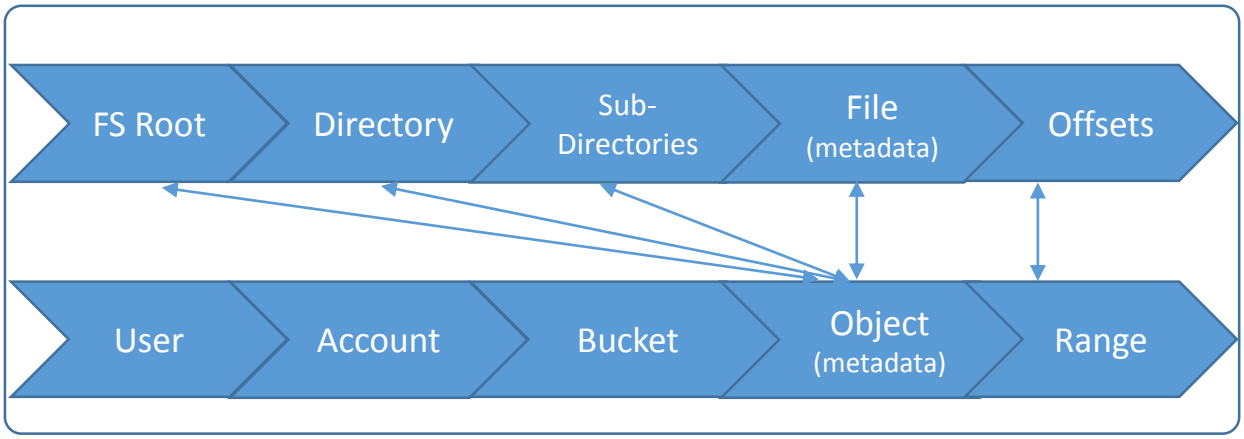
In ECS, the data access layers (services) are decoupled from the data management layers (services). Access protocols are set up as a collection of generic libraries, with multiple service heads (nodes) assigned to serve each file, object, and HDFS protocol.

As illustrated in Figure 2, in this "protocol-agnostic storage at rest" design, ECS treats every file and directory in the file system as a single object. The data and metadata of file and directory are stored in the object created for file and directory. With this, ECS NFS and HDFS easily scale to support billions of files in a single file system. This allows the same storage engine to handle all storage reads and writes.

This also allows all storage services such as metadata, geodispersal, and snapshots to be delivered by a common engine – regardless of the access or ingest protocol. Thus there is no performance overhead or limitations in multiprotocol access – which is becoming quite common these days. It makes ECS extendable to cover other protocols in the future.

FIGURE 2

Protocol-Agnostic Storage at Rest



Source: EMC, 2016

Why It Matters

A problem that plagues unified OBS platforms (with multiprotocol access) or platforms that utilize external gateways for file access is that data ingested via different protocols resides in segregated storage pools. This means that the same policy cannot be consistently enforced across all pools, and change in access requires the data to be manually copied from one location to the other. Similarly, all features and functions cannot be made consistently available across all protocol pools.

ECS therefore works well in environments like analytics and media production where data is frequently ingested via one protocol but accessed via another protocol. In such cases, the size of the data makes it infeasible to move the data around, depending on the choice of access and ingest protocols.

Multiprotocol access and access/ingest-agnostic data at rest make ECS a true enterprise data lake platform (EDLP) — a solution to the data deluge and access conundrum faced by enterprises that cannot be solved using Big Data repositories built on a single file system like Hadoop HDFS. Akin to enterprise data warehouses, EDLPs allow disparate and incoherent data types to be consolidated onto a single, scalable, extensible, and agile storage platform.

Protection Scheme

ECS employs a tiered protection scheme that provides a balance between protecting data locally and globally and maintaining a low dollar-per-gigabyte overhead for data protection. This scheme provides:

- **Local data protection.** ECS tries to protect data with erasure coding (EC). For large objects, data is directly written to EC during ingest. This optimizes the network usage and data ingest performance compared with writing to three copies of data. For EC of 12 + 4 (i.e., 12 data fragments and 4 code fragments), ECS will write 1.33x data to disks, while three copies will write 3x data to disks. For small objects, data is written to three copies of data first and then converted to EC. EC offers the advanced protection at a reduced storage overhead compared with three copies of data. For example, in ECS an EC of 12 + 4 has an overhead of 1.33 and protects against failure of up to 4 disks. Similarly, an EC of 10 + 2 has an overhead of 1.2. For archival storage, an EC of 10 + 2 is recommended, while for important data sets, an EC of 12 + 4 is recommended.
- **Global data protection.** In addition to local data protection, ECS provides global data protection. However, two unique characteristics of ECS are local failure domain handling and a reduction in the at-rest footprint for globally replicated data. In local failure domain handling, even with global protection, it still offers full, uninterrupted local access (when there are site failures). A reduction in at-rest footprint is brought about by implementing an XOR scheme to shrink the secondary copy. ECS offers strong consistency in a geodispersed implementation but falls to an eventually consistent configuration when there is a WAN outage, the caveat being that writes are always returned locally at each site and synchronized when the WAN link is returned. Consistency settings are policy based and can be set based on data ingest at the bucket level.

Why It Matters

ECS overcomes the limitations posed by several early geodispersed architectures. Some common implementations and their drawbacks are:

- **Independent replication and EC schemes at the local and remote sites.** For example, a site locally may implement replicas with an overhead of 3 and EC for replicated data with an overhead of 1.6. When combined, this leads to an overhead of 4.6. This makes the platform highly inefficient from a dollar-per-gigabyte perspective. Many public cloud service providers implement this scheme.
- **Symmetrical, geodispersed architecture.** In this scenario, all datacenters (DCs) are symmetrical and accept writes at all locations. Even if there are a number of DCs, the system can tolerate only a single DC failure. When the system has fully recovered from that failure, then and only then can it tolerate a second DC failure.
- **Combined operational and disaster recovery.** In this scenario, the EC data and parity fragments, or replicas, are distributed across all DCs. For example, if the data is erasure coded with 10 + 6 (10 data and 6 parity fragments), then each DC stores 4 fragments. If one DC goes down, the system reconstructs the fragments among the remaining DCs. The biggest drawback of this architecture is that mixing operational and disaster recovery into a single scheme pushes the overhead onto the WAN, so the penalty is higher WAN bandwidth (and cost) or longer recovery (if the WAN is not fast enough).

ECS uses a hybrid architecture that combines all three schemes. In ECS, the use of an XOR scheme to reduce the "at rest" secondary copy means that the overhead is only a fraction higher than the local overhead. For example, if the local overhead is 1.33 (EC) in a 6 DC configuration, then the combined local remote (one site) overhead is 1.6. When new DCs are added, the overhead rises only incrementally.

In ECS, by default a data chunk is replicated to one of the other DCs for geoprotection, although different chunks might be replicated to different DCs. On a per-chunk basis, each DC decides the other DCs to replicate it. At a given time, any DC can be asked for the primary copy of the data or a backup copy. Each DC in ECS does XOR among the secondary chunks georeplicated to the DC from other DCs. For example, in a 3 DC system, chunk A's primary DC is DC1, and it is georeplicated to DC3; similarly, chunk B's primary DC is DC2, and it is also georeplicated to DC3; and then in DC3, XOR will be done among the secondary of chunk A and B, which saves the overhead of storing the secondary from 1.33 to 0.66. When a DC goes down, the other DCs use the remote XOR and local chunk fragments to reconstruct the lost copies. Once these copies are recovered, that DC assumes primary ownership of this copy.

The biggest benefit of this scheme is that when there is a local failure (i.e., within the DC), the recovery is performed locally, and there is no WAN overhead. When there is a DC failure, the remote recovery is based on a global XOR, with minimal WAN overhead to replicate changed data. This XOR mechanism for protecting chunk applies to user data. The system data, which is very small compared with user data, will be replicated to all DCs in the same Replication Group.

This scheme also eliminates the burden of choosing the primary location and/or choosing the best location for access from the application. The system internally balances the data and provides concurrent read/write access across all DCs. The application gets a consistent view across all DCs. An application never gets an object-not-found error.

If an object is not available locally, it is fetched over the WAN. In the background, the corresponding data chunk will be cached locally. So the next read will be served from the local cache.

With writes, the data is updated asynchronously. Object metadata georeplication is asynchronous as well, though updates to bucket and namespace are georeplicated in sync while metadata is updated synchronously.

FUTURE OUTLOOK

As businesses enter the 3rd Platform era, data will become their most valuable asset — and next-generation applications will become the vehicle via which this data is generated, consumed, and delivered. Traditional infrastructure designed before the cloud computing era has given rise to data silos and added complexity while going global. This infrastructure is not set up to host next-generation applications.

NGAs will continue to perpetuate a need for geographically dispersed data creation and consumption via mobile devices, M2M, and IoT ecosystems. The IoT ecosystem continues to grow rapidly, with over 30 billion devices connected to the Internet by 2017. These "connected devices" that include wearables, sensors, and telemetry devices on jet engines can produce terabytes of data in just one hour. This data is created only once and can be analyzed in various and multiple ways, bringing up the importance of storage in IoT scenarios.

NGAs generate data at rates that are significantly higher than those of "traditional" applications like databases, email, and office productivity applications. They are designed with in-place real-time analytics and a user-first experience – which means the conversion from data to information has to happen with minimal movement.

All of these attributes underscore the need for OBS platforms like ECS to form the underpinning of infrastructure for NGAs. However, all OBS platforms that offer standardized APIs like Amazon S3 and OpenStack Swift are not created equal – in fact, quite the contrary, and hence the need for vendors to adopt a clean-slate approach to designing exascale OBS platforms.

CHALLENGES/OPPORTUNITIES

IDC believes that during the design process for ECS, EMC took the bold but tedious step of understanding the limitations of:

- Public cloud OBS platforms from vendors such as Google, Microsoft, and Amazon
- Most OBS platforms in the market today
- Its own OBS platforms such as Atmos and Centera

The result is that the market could perceive EMC as a late entrant – and fail to fully appreciate the key differentiators of ECS and why the platform deserves to be evaluated as part of NGA infrastructure efforts.

This presents challenges and opportunities for EMC. The company needs to catch up with its OBS competitors in terms of messaging, mindshare, and market perception. EMC also needs to show an undeterred commitment to the OBS market by way of successful wins. In parallel, it needs to aggressively build out an ISV ecosystem – partners that can certify ECS as an OBS platform for their NGAs. Given EMC's might and past successes with the Centera and Atmos offerings, IDC believes that this is not a question of "if" but a question of "when."

CONCLUSION

EMC has taken a "from the ground up" approach in designing a next-generation OBS platform. With features and functions (notably scale) for IDC's 3rd Platform era apps, EMC is poised to regain a leadership position in the OBS market.

About IDC

International Data Corporation (IDC) is the premier global provider of market intelligence, advisory services, and events for the information technology, telecommunications and consumer technology markets. IDC helps IT professionals, business executives, and the investment community make fact-based decisions on technology purchases and business strategy. More than 1,100 IDC analysts provide global, regional, and local expertise on technology and industry opportunities and trends in over 110 countries worldwide. For 50 years, IDC has provided strategic insights to help our clients achieve their key business objectives. IDC is a subsidiary of IDG, the world's leading technology media, research, and events company.

Global Headquarters

5 Speen Street
Framingham, MA 01701
USA
508.872.8200
Twitter: @IDC
idc-community.com
www.idc.com