APPLICATION MODERNIZATION WITH DELL EMC ELASTIC CLOUD STORAGE (ECS)

Reducing costs and improving responsiveness to the business

ABSTRACT

This white paper explains how ECS can help organizations improve existing applications by using storage more effectively.

December 2016

TABLE OF CONTENTS

EXECUTIVE OVERVIEW

Like people, applications need to adapt as the world changes. Eventually, almost every application needs to be modernized. For applications that use large amounts of unstructured data, an effective way to do this is to change how the application uses storage. In place of traditional network-attached storage (NAS), the application can instead use a modern object store like DELL EMC® Elastic Cloud Storage (ECS™) or a scale-out NAS system such as DELL EMC Isilon™. This paper focuses on the benefits of using modern object stores such as DELL EMC ECS.

Moving an application from traditional NAS to ECS can provide many benefits, including the following:

- ECS provides an efficient object store that can be shared by many applications and groups within an organization. This built-in multi-tenancy makes it easy to provide a shared storage utility.
- ECS lets applications access massive amounts of data through a single global namespace. Rather than worrying about where to find a particular file, developers just supply an object name and let ECS handle the rest.
- ECS provides built-in backup and disaster recovery through geo-replication of data. And because multiple copies of data can be stored in different places around the world, applications anywhere can access that data quickly.
- ECS allows direct HTTP access to data via the Internet. It also provides searchable metadata attached to every object, making it easier for applications to find that data they need.
- ECS allows developers to provision the storage they need when they need it without waiting for storage administrators. This minimizes delays and improves time to market for new applications.

Getting these benefits requires modifying an application to use ECS rather than traditional NAS. Yet doing this typically takes only days or weeks of work, an investment that's quickly paid back. If you're looking to modernize existing applications that use lots of unstructured data, you need to consider ECS.

THE CHALLENGE: MODERNIZING APPLICATIONS

No matter how good your custom applications are, your users always want more. Maybe they want the application to deliver content over the internet more effectively, for example, or maybe they just want it to store more data. And IT leaders like you also want to make your applications better. You might want to make them easier to change, for instance, and you'd almost certainly like to make them cheaper to run.

To meet those needs, you have to improve—modernize—your applications. One way to do this is to throw the current code away and start over. This is sometimes the only viable option, but it's expensive and time-consuming. A better solution is to modify the application just enough to meet the new requirements, ideally in a way that prepares the codebase to meet future demands.

For custom applications that use large amounts of unstructured data, the best way to do this is often to change how the application uses storage. This is true whether it's a document management system, or an application that stores images of checks, or a security surveillance system storing many hours of video, or something else. Rather than relying on the traditional network-attached storage (NAS) provided by NFS fileservers, you can modify the application to use a more modern approach, such as using an object storage platform.

DELL EMC Elastic Cloud Storage (ECS) can help you achieve these goals. In place of a traditional file system, ECS provides an object store, a more modern approach to working with unstructured data. By changing only how your applications store and access this kind of data, you can make life better for your users, your development team, and your infrastructure staff. Figure 1 illustrates the idea.

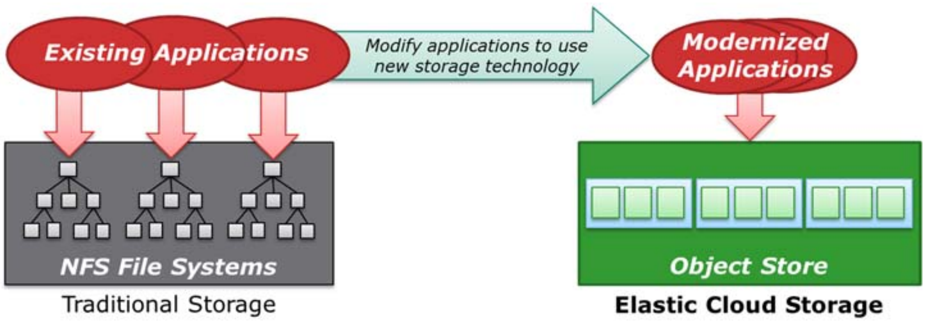

Figure 1. Existing custom applications that use large amounts of unstructured data can be significantly improved by modifying them to use an ECS object store.

To see why doing this makes sense, we need to look first at the limitations inherent in traditional NAS

The Limitations of Traditional NAS

Traditional NAS systems today have a number of challenges. These challenges can be grouped into two categories: high costs and slow responsiveness to changing business needs. It's worth looking at each category separately.

High costs

Several aspects of traditional NAS lead to unnecessarily high costs. Figure 2 illustrates the most important of these.

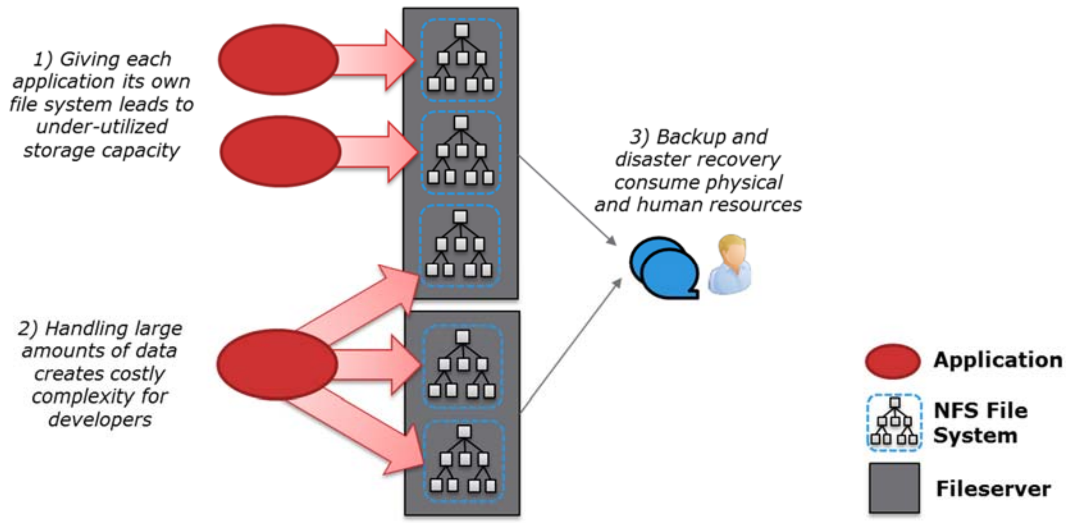

Figure 2. Traditional NAS systems have high costs, especially for applications that work with lots of data.

As the figure shows, traditional NAS characteristics that lead to high costs include the following:

1. **Under-utilized storage capacity.** It's common to create a separate NFS file system for each development team or even each application. While this helps ensure data isolation, it also leads to under-utilized storage. In some cases, as much as half of storage capacity can sit unused.

2. **Increased complexity at scale.** NFS has real limitations for applications that need to store terabytes or petabytes of data. This is especially true if this data is spread across a very large number of files. A common way to deal with this is by storing an application's data in multiple file systems. Yet doing this can require developers to know which file system holds the data they need. Alternatively, the development team might create a storage abstraction layer that provides a common namespace across multiple file systems. In either case, the result is more complexity, more development time, and higher costs.

3. **The need for separate backup and disaster recovery solutions.** Traditional NAS systems need regular backups, along with a way to restore backed-up data when necessary. This requires paying for people and more storage. Traditional NAS systems also commonly need to have disaster recovery mechanisms in place, adding to the expense.

Slow Responsiveness to Changing Business Needs

Along with their high costs, traditional NAS systems also make it harder for your development teams to respond quickly to changing business needs. Figure 3 illustrates these problems.

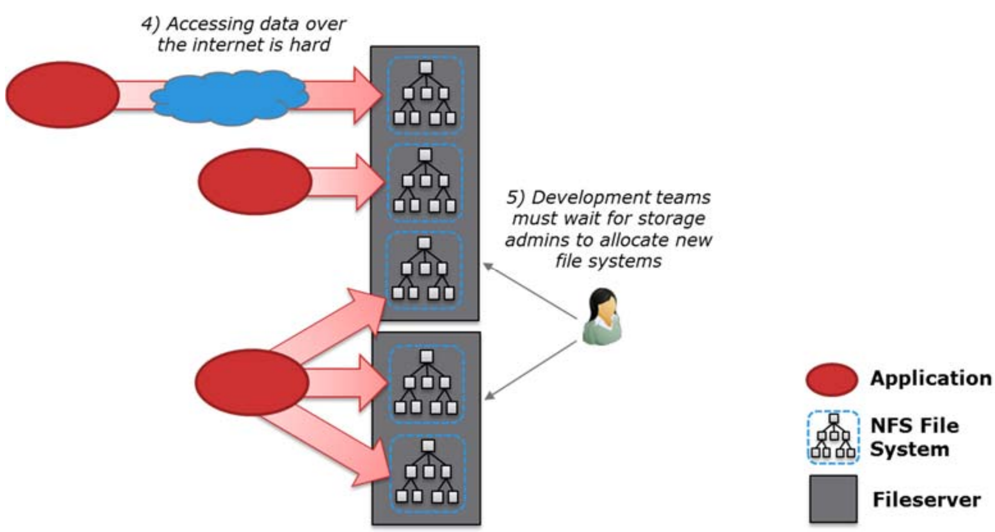

Figure 3. Traditional NAS systems make it harder to respond quickly to changing business needs.

As the figure shows, the problems include the following:

4. **Difficulty in** delivering **content over the Internet.** Suppose the business decides that a mobile client is an essential addition to an existing application. To ensure good performance, this application needs direct HTTP access to data. Providing this with traditional NAS is challenging.

5. **Longer provisioning time.** With traditional NFS systems, storage administrators must allocate new file systems. This can result in delays in the development process, which in turn slow down a team's ability to release new software.

What ECS Provides

Converting your existing custom applications to use ECS can alleviate all of these problems. Figure 4 illustrates how ECS eliminates the traditional NAS limitations just described.

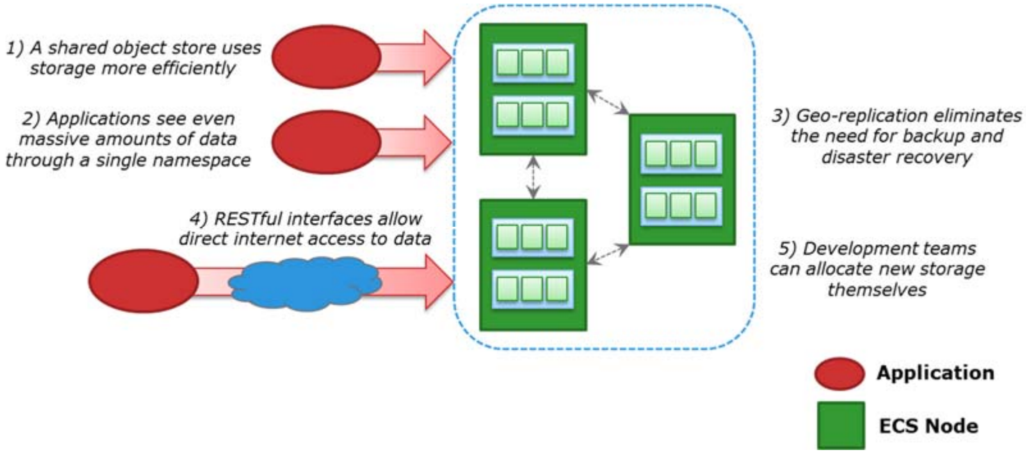

Figure 4. ECS lowers costs and increases responsiveness by addressing the challenges of traditional NAS systems.

Here's how ECS addresses each of these five limitations:

1. **Efficient shared object store.** Rather than requiring separate file systems for each application, ECS is designed to be a multi-tenant service. It provides a storage utility that can be effectively shared by many applications. This eliminates the unused capacity that NFS storage silos commonly create.

2. **Simple global namespace for massive amounts of data.** Because ECS is an object store, it allows access to hundreds of petabytes of data and billions of objects, going well beyond the limitations of NFS. And applications see all of this data as a single namespace, even if the data is spread across multiple physical nodes. This eliminates the complexity—and the costs—of forcing developers to deal with diverse NFS file systems.

3. **Built-in backup and disaster recovery.** ECS provides geo-replication of data across multiple nodes. These nodes can be spread across a city, a country, or the entire world, and ECS will keep data synchronized across them. Because of this, the money you spend today for traditional NAS backup and disaster recovery can be eliminated.

4. **Direct internet access to data via HTTP.** This lets mobile applications and others go right to the information they need without opening a new port in your firewall. ECS also provides searchable metadata attached to every object, making it easier for applications to find the right data quickly.

5. **Fast and easy provisioning.** Rather than waiting for storage administrators, ECS allows developers to provision the storage they need when they need it. The result is that development teams can create and deploy new applications more quickly.

To get these benefits, you must modify your applications to use the ECS object store rather than NFS. Doing this isn't conceptually difficult; ECS provides a RESTful interface with GETs and PUTs that substitute for reads and writes. The effort required can range from a few days to a few months, depending on the application. Still, especially for applications that use large amounts of unstructured data, the payback can be substantial.

ECS: A MODERN STORAGE PLATFORM

ECS offers a way to improve your applications by changing only how they access storage. The technology also supports other use cases, such as archiving and creating new cloud-scale applications. The focus here, however, is on the benefits that ECS provides for application modernization. What follows takes a closer look at this technology, describing what it is and how it provides the benefits just described.

ECS is available as either an appliance, with hardware supplied by DELL EMC, or software that runs on your own industry standard hardware. In either case, the technology provides the same fundamental services, as described next.

Storing Data: Objects and Buckets

Because ECS is an object store, the place to start is by examining objects. While DELL EMC arguably invented the concept of an object store, other vendors also provide this storage style today, and the basic concepts are similar across all of them.

First, note that objects in an object store have nothing to do with objects in programming languages such as Java or Python. Instead, an object in this context refers to a storage abstraction, as Figure 5 shows

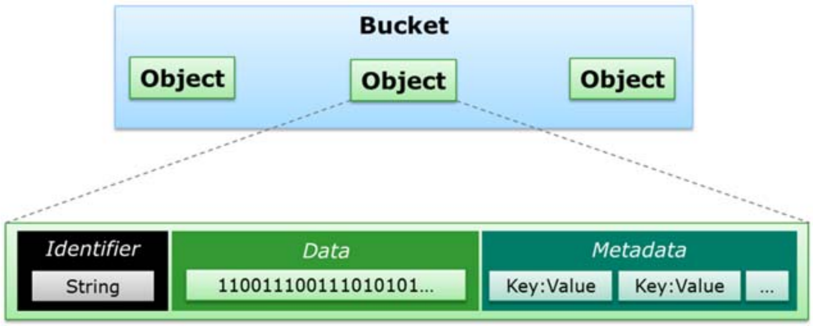

Figure 5. Every object has three parts and is contained in some bucket.

Every object is contained in a bucket, and each bucket has a globally unique name. Unlike a traditional file system, however, there's no further hierarchy: Buckets can contain only objects, not other buckets.

As the figure shows, each object has three parts:

- An identifier, which is a string that uniquely identifies an object within its bucket.
- Data, which is just binary information. In ECS, an object's data can be as small as a single byte or as large as many petabytes. If desired, ECS can encrypt this data as well.
- Metadata, which is a set of key/value pairs. Some parts of an object's metadata, such as size and creation date, are defined by the system. You can also create your own metadata for each object, such as AccountID:33445 or DocumentType:Invoice.

To access a particular object, an application can provide the object's bucket name and object identifier. Alternatively, an application can retrieve objects by using the ECS metadata query language to search object metadata. For example, an application might search for all objects with DocumentType:Invoice that were created on a specific date. This obviates the need for a separate database to contain the metadata associated with these objects.

ECS also provides optional versioning for all objects in a bucket. When this feature is turned on, ECS will store each change to an object, including deletion of the entire object, as a new version. If an object is accidentally changed or deleted, you can restore an earlier version to avoid losing data.

To protect against failures, ECS uses erasure coding, a technique that stores redundant information with each object to allow recreating data that's lost or corrupted. Each object is then spread across multiple disks. If one or more disks fail, the lost information can be recreated from the object's redundant information. This technique protects against hardware failures without the overhead of maintaining multiple copies of each object.

Organizing and Accessing Data

Buckets, along with the objects they contain, are grouped into namespaces. Applications can access these namespaces through several different protocols, as Figure 6 shows.

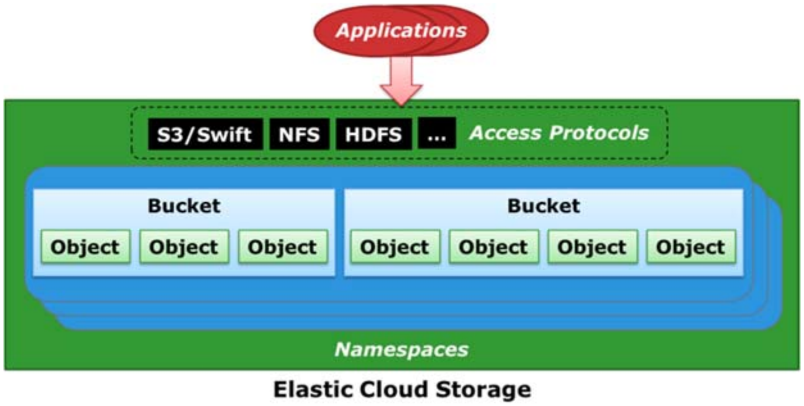

Figure 6. Applications can use various protocols to access the objects and buckets within a namespace.

Namespaces

A namespace provides a way to organize and control access to a group of related buckets and objects. A single ECS environment can have many namespaces, which means that multi-tenancy is built in. In other words, ECS is designed to provide a shared utility for everyone in your organization to use. Each department in your organization or even each application might be assigned its own namespace.

Don't expect to create multiple namespaces just for scale, however; you won't need to. There are two different aspects of scale for data that might matter to an application: handling large amounts of data and handling large numbers of files. With NFS, for example, a volume might hold between 2 and 64 terabytes of data, and stuffing more than 10,000 files into a single directory is problematic. With ECS, a single namespace can contain hundreds of petabytes of data with billions and billions of objects. The limits of traditional NFS systems go away.

Another important difference from traditional file systems is that development teams can provision buckets themselves within their namespace. While ECS lets IT administrators set quotas for buckets and namespaces, there's no need to wait for those administrators to make new storage available. This lets developers get their improved applications up and running significantly faster.

Access Protocols

As Figure 6 shows, applications can access ECS data through several different access protocols. The options include the following:

- RESTful protocols defined by Amazon Web Service's Simple Storage Service (S3) and OpenStack's Swift. These protocols provide straightforward access to data using HTTP verbs such as GET and PUT. They also allow direct access to objects, since firewalls typically allow HTTP access.
- The standard NFS protocol.
- The access protocol defined by the Hadoop Distributed File System (HDFS). This protocol is rapidly becoming the default for accessing the diverse data stored in data lakes.
- DELL EMC-specific protocols, including the protocol used with DELL EMC's earlier object store Atmos.

Of the various protocol choices ECS provides, S3 is by a wide margin the most commonly used. In most cases, modernizing an application to use ECS will mean replacing the NFS calls it currently uses to access data with the analogous S3 operations.

But since ECS supports the NFS protocol, mapping the NFS file structure onto objects, why do you need to change your application at all? Why not just move it as is to ECS using the product's NFS support? In fact, you can do this. However, ECS is an object store, not a file server, and so access through NFS will be significantly slower than access through S3. In general, think of the NFS interface as a helpful way to migrate an application to the native object store access provided by S3, not as a replacement for an NFS fileserver.

In fact, even through S3, access to data in ECS will have more latency than access to data in a traditional NFS file server. This is the nature of object stores, and it means that ECS isn't the best choice for applications that can't tolerate data access times in the tens of milliseconds. For example, an object store like ECS isn't the best option for running a database management system or providing swap space for an operating system.

It's important to realize that the same ECS data can be accessed through different protocols. One important example of this is that information your applications write through S3 can be read via HDFS. This makes it easier for big data tools such as those in the Hadoop family to access your data for analysis.

Geo-Replicating Data

Many of today's applications were created with the assumption that the application and its data are in the same datacenter. Making this assumption simplifies life for the application's creators, and it also simplifies the storage requirements.

Yet a common request that users make for application modernization today is to provide fast access to applications and their data from any device anyplace in the world. Doing this with traditional fileservers can get complicated, especially for applications that work with large amounts of data.

ECS is designed to address this problem. Rather than making you create multiple file systems spread around the world, then handling replication yourself, ECS automatically replicates data across nodes. Figure 7 illustrates this idea.

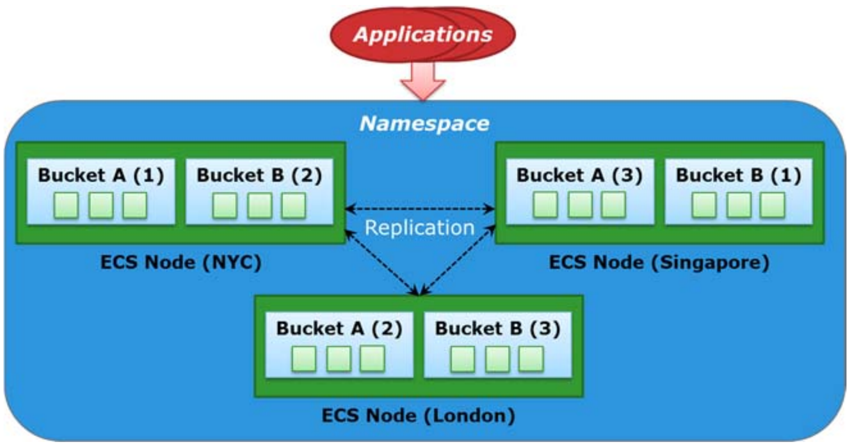

Figure 7. ECS replicates buckets and the objects they contain across nodes, giving applications fast access to data wherever they are.

In the example shown here, an organization has deployed three ECS nodes: one in New York City, another in London, and a third in Singapore. All three contain copies of the same buckets and the objects in those buckets. All of these copies are read/write, so applications running in any of these locations can have fast access to data; there's no need to rely on slow links to a single datacenter somewhere in the world.

ECS automatically keeps all of the copies in sync, making sure that applications always see the most current data. And an ECS administrator can define exactly how replication is done. If a bucket contains data that can't leave the EU, for example, the administrator can make sure that ECS replication never violates this constraint.

Another benefit of ECS's built-in geo-replication is that it removes the need for data backups and disaster recovery (DR). Since consistent copies of the data already exist in multiple locations, why spend extra money on these things? And remember that ECS also provides object versioning, protecting against accidental deletion, along with erasure coding. In a geo-replicated ECS system, backups and DR are redundant. This can save significant amounts of money compared to the costs of backup and DR for traditional NFS file systems.

It's also important to realize that, as Figure 7 suggests, applications see the data in geo-replicated ECS nodes as a single namespace. Developers don't need to work out which copy is closest to an application at any particular time, nor is there a need to write a storage abstraction layer that makes different fileservers appear as a unified whole. Instead, the application just provides the name of an object or bucket, and ECS finds the best available node that contains that data. This is especially useful with mobile applications, where the same URL can potentially provide fast access to data from locations around the world.

Geo-replication also helps applications handle more users. Rather than direct all reads and writes to a single ECS node, those requests can be load balanced across multiple nodes. Since the load is now spread across several different machines, the application as a whole can provide faster access to more users.

Using Containers with ECS

As part of a modernization effort, applications are often modified to use containers. The adoption of containers is driven by two key benefits. First, containers can help loosen the coupling between components in applications, as well as between the application and the operating system. This makes these applications easier to deploy, upgrade and scale. Second, containers provide greater flexibility for applications to use different pools of infrastructure resources, thereby enabling easy relocation or scaling using new pools of infrastructure.

Application developers don't need to do anything special to access ECS from within a container. In fact, due to the advantages noted earlier in this whitepaper, ECS brings the benefits of containerization, including loose coupling and easy scaling, to the storage layer.

SUMMARY

ECS is a powerful technology. Using ECS to modernize applications by replacing NFS file systems has many benefits, especially when those applications use lots of unstructured data. Those benefits include the following:

- ECS provides a shared object store that avoids allocating separate file systems to each application and the under-utilization of storage this implies.
- ECS gives applications access to very large amounts of data through a single namespace while hiding this complexity from developers.
- ECS eliminates the need for backup and disaster recovery by providing features such as geo-replication, object versioning, and erasure coding.
- ECS lets applications access their data directly via HTTP with good performance from around the world.
- ECS lets development teams move faster by allowing self-service allocation of new storage.

If you're looking to lower the costs of running your applications while also improving your ability to respond quickly to changing business needs, ECS has a lot to offer.

To learn more about the DELL EMC ECS storage solution, please visit http://www.emc.com/en-us/storage/ecs/index.htm