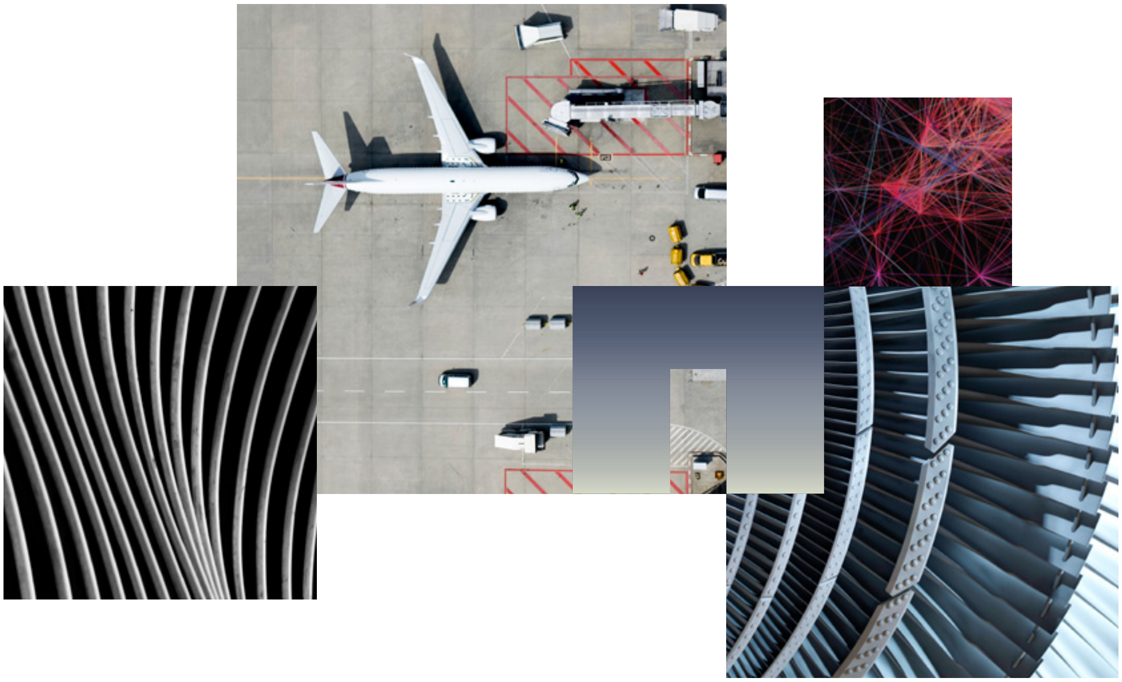

WHITE PAPER

Amazon EFS: Implementation in the Real World

NetApp®

Executive Summary

This white paper post looks at considerations for a real-world deployment of Amazon Elastic File System (Amazon EFS). Though getting started with EFS is simple, to use it as a production platform, you need to consider several factors, such as backups and snapshots. Amazon EFS does not currently provide a system for creating snapshots or backups, which the end user must implement. There are several ways to achieve this, and we examine some of them here. Another area of consideration is performance, which can vary in relation to the amount of capacity being used.

Introduction

Amazon Elastic File System (EFS) greatly simplifies the setup of a horizontally scalable cloud-based file system, allowing you to access the same files concurrently from hundreds, and even thousands, of different clients. Because it's a fully managed solution, getting started is very easy. The wizard interface allows you to create a new, highly available file system in seconds, with all files redundantly stored across Availability Zones. Amazon EC2 Linux hosts can then use standard commands to connect to the file system over NFS.

You can use Amazon EFS to grant access to your data for other cloud services, such as big data analytics. It can also support the shared data required by web applications and content management systems, or serve as a backup location for on-premises file servers.

However, as with all new platforms, it is essential to test and evaluate Amazon EFS to ensure that it will work in your production environment. In this paper, we look at some of the typical implementation challenges you might face using Amazon EFS and ways of overcoming them. If you're still researching Amazon EFS, this paper will give you some useful insights ahead of time, and details an alternative solution for creating cloud-based file systems by using NetApp® Cloud Volumes Service.

On-Premises Access

Access to Amazon EFS file systems is intended primarily for use by Amazon EC2 instances, which can simply mount the file system to a directory by using the Linux `mount` command. However, in several use cases, access to Amazon EFS is required from on-premises systems. This might be to load data from an initial set of files to be processed or to access the results of work that has already been performed.

Directly mounting an Amazon EFS file system to an on-premises server requires the use of AWS Direct Connect, which is used to create a private, dedicated network connection directly to AEPRthe AWS cloud, removing the need to go over the internet. Direct Connect comes with many benefits, including more consistent network performance and bandwidth efficiency. But it requires you either to co-locate your systems in an existing AWS Direct Connect location or to work with an Amazon APN Partner to set up connectivity. Note that access to Amazon EFS over Amazon VPN is not supported, and that use of AWS Direct Connect will incur its own charges.

An alternative to AWS Direct Connect is to use Secure Copy Protocol (SCP) from your on-premises server to an Amazon EC2 instance that has the Amazon EFS file system mounted to it. This approach makes it possible to copy files to and from the Amazon EFS file system.

Backup and Recovery

Amazon EFS hosts a live, shared file system. If you accidentally delete files or overwrite files incorrectly, you will need to revert to a backup. Unlike Amazon Elastic Block Store (Amazon EBS), Amazon EFS cannot create NetApp Snapshot™ copies, which are read-only, point-in-time images of a storage volume. There is also no integrated backup capability, which means that making backups of an Amazon EFS file system is largely a manual process.

Backup and Restore with AWS Data Pipeline

Amazon provides AWS Data Pipeline, a solution that can be accessed on GitHub. Data Pipeline works by instantiating an Amazon EC2 instance to connect to the primary Amazon EFS file system and to a secondary, backup Amazon EFS file system. The solution then uses the standard Linux `rsync` utility to synchronize data between the file systems. This synchronization ensures that only files that have changed are included in later backups. The shell script used in AWS Data Pipeline also takes care of other considerations such as backup rotation. A separate Data Pipeline allows you to restore a previous backup.

Another feature of the Data Pipeline approach is the ability to use multiple nodes to complete a backup faster. Each node backs up a different portion of the files to the other nodes. For example, in a two-node configuration, the first node would back up all odd-numbered files and the second node all even-numbered files.

There are several considerations to keep in mind when using this approach:

- **Manual process.** All the tasks involved with setting up AWS Data Pipeline for backup and restore, including the setup of the destination Amazon EFS file system, must be performed by the end user.
- **Online backup.** When you create Amazon EFS backups, the copy is performed while the file system is still active, so changes to files that the backup process has not reached yet might affect the consistency of the overall backup. Usually, these backups are done during off-peak hours or times of low activity.
- **AWS resource usage.** AWS Data Pipeline will manage the lifecycle of any Amazon EC2 instances that the backup process uses. There are also costs associated with using AWS Data Pipeline itself, as well as costs associated with the backup Amazon EFS location.

- **Throughput usage.** Because the backup process is just like any other process accessing the Amazon EFS file system, the throughput used counts toward overall throughput usage, which, as we shall see later, could affect the performance of your file system.

Alternatives

If you don't require concurrent access to your backup files, it might be cheaper to mount an Amazon EBS disk, such as Cold HDD (sc1), to one of the Amazon EC2 nodes that already accesses your Amazon EFS file system. You can then use a process based around `rsync` to achieve most of the benefits of the AWS Data Pipeline approach, but at a tenth of the cloud storage costs. A multi-node backup solution, however, would be more complicated to achieve.

Another alternative is to use the AWS CLI utility to incrementally synchronize the Amazon EFS file system with Amazon Simple Storage Service (Amazon S3), which provides one of the most cost-effective solutions for cloud storage. This process works best when you require only one up-to-date backup to be maintained, because the incremental updates would be applied to a single Amazon S3 destination. If further, separate backups are required, each one would need its own full copy of the Amazon EFS file system, which is something the `rsync` approach handles more efficiently.

Amazon EFS Performance

Amazon strongly recommends using Amazon EC2 instances with a Linux distribution that runs a kernel version of 4.0+, because this approach can have a significant impact on Amazon EFS file system performance. It's also important to correctly specify NFS version 4.1 and the read/write size options in the `mount` command, according to the documentation.

Performance Modes

Amazon EFS supports two performance modes: General Purpose and Max I/O. You must select one of these modes when you create a file system; the default is General Purpose. The Max I/O mode trades higher file access latency for increased I/O throughput and is typically used when a large number of EC2 instances need access to the file system at the same time. To determine whether you will benefit from Max I/O, you can create your file system by using General Purpose and monitor the PercentIOLimit metric in Amazon CloudWatch.

Write Latency

Amazon EFS provides a strong guarantee of consistency for write operations. This consistency means that when data is written to an Amazon EFS file system, an acknowledgment will not be returned to the client performing the write until it has been replicated to more than one Availability Zone. This approach creates a certain amount of latency for each file write operation, especially if the operations are performed in serial—that is, one after another. One workaround is to increase the size of the I/O operations being performed, which essentially increases total throughput per operation; however, this workaround isn't possible for small files.

Another way to increase throughput is to parallelize writes to the Amazon EFS file system across multiple threads and Amazon EC2 instances. Throughput increases in proportion to parallelism consistently, so the choice of Amazon EC2 instance type can also play a part in determining total aggregate performance. You can use tools such as GNU parallel to orchestrate parallel file copies.

Capacity and Throughput Relationship

When working with Amazon EFS, it's very important to understand how throughput is affected by the amount of storage your file system has allocated. Each Amazon EFS file system has an aggregate baseline throughput of 50KiBps for every gibibyte (GiB) of allocated storage. You can achieve higher rates of throughput by earning burst credits, which accumulate at the same baseline rate during periods of inactivity. Burst credits allow you to achieve throughput rates of up to 100MiBps per tebibyte (TiB) of allocated storage, and are consumed when you read or write data to the file system. When all burst credits have been consumed, file system throughput drops back to the baseline rate.

So, for example, a 10GiB file system on Amazon EFS would receive a baseline throughput rate of 0.5MiBps. Because the rate at which burst credits are accumulated would also be 0.5MiBps, you would be able to achieve the maximum burst performance of 100MiBps for 0.5% of each day, which comes to 7.2 minutes. Scaling up to 1TiB of storage would allow you to burst for 50% of each day, which is the maximum.

If you have a small, actively used file system, this can create significant performance problems, which you can remedy only by allocating more storage, increasing your cloud storage costs. These performance problems might not be noticeable when you first create an Amazon EFS share, because all new shares come with a fairly large balance of burst credits to make it easier to get data into Amazon EFS. The best way to tackle this challenge is to calculate in advance the amount of storage you'll need to allocate to guarantee your minimum level of required bandwidth.

An Alternative Approach: Cloud Volumes Service

Another solution for creating cloud-based file shares in AWS is NetApp Cloud Volumes Service, which is a fully managed, highly available, high-performance cloud-native NAS solution, supporting both NFS and SMB. As a native cloud solution, Cloud Volumes Service is available in AWS just like any other cloud service, such as Amazon EFS, which means you can start using it without any prior installation or setup.

Using Cloud Volumes Service, you can create new shared file systems instantly, and can quickly scale up capacity at any time. In addition to NFSv4, Cloud Volumes Service supports NFSv3 and SMB with Active Directory integration. You can configure performance when creating a Cloud Volumes Service file system by choosing an appropriate service level, which determines the level of IOPS the storage will deliver.

Cloud Volumes Service automatically provides data redundancy and high availability across Availability Zones. Additionally, Cloud Volumes Service supports instant Snapshot copies that can be used to protect data against accidental changes or other such issues. You can also use Snapshot copies to instantly create writable clones of an active file system, which can be used for testing purposes, for example. To help onboard new systems, Cloud Volumes Service comes with built-in synchronization services that can efficiently and incrementally copy data from on-premises environments, or other cloud systems, into or out of Cloud Volumes Service.

Conclusion

As we've seen in this article, there are several things to consider when choosing Amazon EFS for production file system storage. Some of these considerations require an in-depth knowledge of Linux tools and the ability to manually set up processes, such as for backup and restore. Yet other considerations require planning, estimation, and ultimately testing.

For those still weighing their options, Cloud Volumes Service provides a ready-to-use solution for NFS in the cloud, in the same as Amazon EFS. With decades of experience in building on-premises NAS storage systems, NetApp delivers an enterprise-grade NFS solution with Cloud Volumes Service, with additional features for creating Snapshot copies, cloning, and synchronizing data across systems.

To see if NetApp Cloud Volumes Service is right for you, sign up to try Cloud Volumes Service today.

About NetApp

NetApp is the data authority for hybrid cloud. We provide a full range of hybrid cloud data services that simplify management of applications and data across cloud and on-premises environments to accelerate digital transformation. Together with our partners, we empower global organizations to unleash the full potential of their data to expand customer touchpoints, foster greater innovation, and optimize their operations. For more information, visit www.netapp.com. #DataDriven

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

WP-7278-1018

NetApp®