WHITEPAPER

SwiftStack Filesystem Gateway Architecture

March 2015

by Amanda Plimpton

Executive Summary

SwiftStack's Filesystem Gateway expands the functionality of an organization's SwiftStack deployment by exposing object-stored data as files and providing integration with file-based applications. This gives organizations the benefits of SwiftStack's object storage, including single-pane management tools, while simultaneously leveraging legacy data storage and interfaces. Having the ability to bridge the two data types creates the flexibility needed for modern storage solutions to meet archiving, disaster recovery, and migration needs of an organization. This unified storage solution has the business agility, cost efficiency, enterprise resiliency and scalable architecture needed to position it as the obvious storage solution of choice.

Audience

This whitepaper is written for a technical audience that is aware of the power of object storage. As architects and visionaries that understand the requirements of this new paradigm in storage, this audience is trying to figure out how to support legacy file-based applications while upgrading to a more scalable, distributed and moreover low cost storage architecture. This audience will learn how SwiftStack can help to unify their object and file-based storage with the SwiftStack software defined storage platform.

About the Author

Amanda Plimpton is a wearer of many (stylish) hats at SwiftStack. She brings ten years of diverse IT experience, plus a dreadful sense of humor, to the team and is fascinated by the various aspects of the company and business sector she works in. One of the joys of her current position is being able to pick the brains of her brilliant colleagues and translate the exciting work they are doing into consumable media for everyone to enjoy. She knows data storage isn't a compelling topic to everyone — and is working hard to change that.

SwiftStack

Introduction

As the need to store data continues to grow, so does the need for flexible storage options that allow organizations to migrate legacy data to the new industry standard of object storage without disrupting business processes or incurring costs related to converting legacy applications. The SwiftStack Filesystem Gateway (the Gateway) provides that flexibility.

The Gateway allows organizations with a need for SMB/CIFS or NFS file-based access to take advantage of SwiftStack's scalability, durability, availability and low cost for object storage. In addition to the benefits of using an object storage system, the Gateway saves time and money for organizations because it has no hardware vendor lock-in and integrates with existing file-based applications.

By leveraging the Gateway, a SwiftStack Cluster provides the best solution for a number of use cases:

- Active Archiving
- Disaster Recovery
- Content Distribution and Management
- Migration Services
    - Legacy data to object storage for use with existing legacy applications
    - Legacy data to object storage for use in new web-based applications
    - Large data sets for analytics

Business Agility

The Gateway makes it possible to unify file and object-based storage in a single solution that permits data migration from legacy file storage to object storage without impeding access to either storage systems.

In addition, SwiftStack object storage can be accessed using the Gateway or through the Swift HTTP RESTful API, so the same data can be accessed as either files or objects.

Having a unified storage platform with a SwiftStack deployment gives organizations versatility with data storage and access, without vendor lock-in or reliance on moribund systems.

Cost Efficiency

The Gateway runs on low-cost, commodity hardware and installs on standard Linux server distributions. There is no vendor lock-in on the hardware or on data stored via the Gateway. Instead your organization provisions exactly the hardware it needs, when it needs it, to achieve the best performance at the best price. Similarly any data stored by the Gateway is fully and equally accessible through the native Swift API allowing your organization to use the most cost-effective access method for individual applications.

Enterprise Resiliency

The Gateway, like the other components of a SwiftStack deployment, is resilient with non-disruptive upgrades, seamless integration with enterprise authentication such as LDAP/AD to preserve file ownership and permissions. With proper configuration the Gateway can be fault-tolerant.

Scalable Architecture

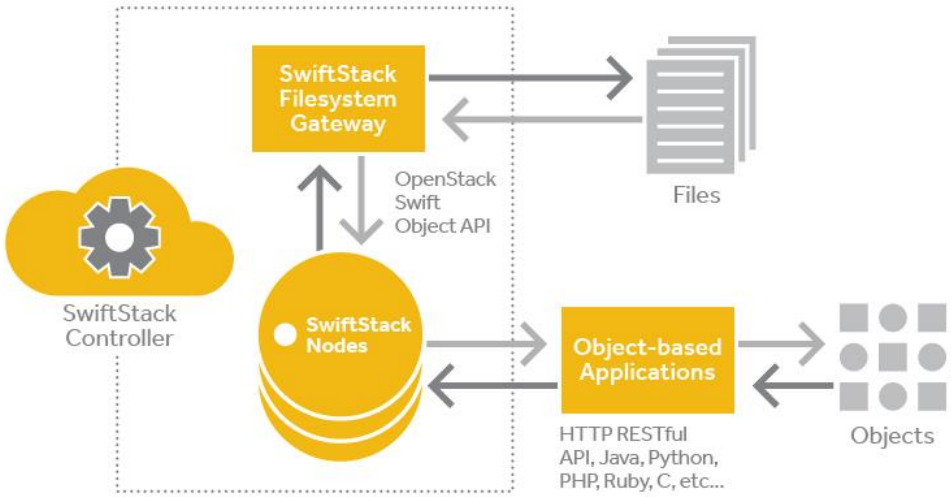
Multiple Gateway nodes can be connected to the same SwiftStack cluster to scale up performance in cases where an increased demand on a single Gateway begins to impact it. The data accessed by the Gateway is stored in a SwiftStack Cluster, so it can easily scale out from a few terabytes to hundreds of petabytes to meet increased need for data storage.

System Overview

General

The Gateway is a component that is added to an existing SwiftStack deployment to provide an additional way of accessing objects using network filesystem protocols.

The Gateway is ingested by the SwiftStack Controller and linked to the desired Cluster by administrators who then use the Controller's web interface to configure the Gateway and set the filesystem mount points.



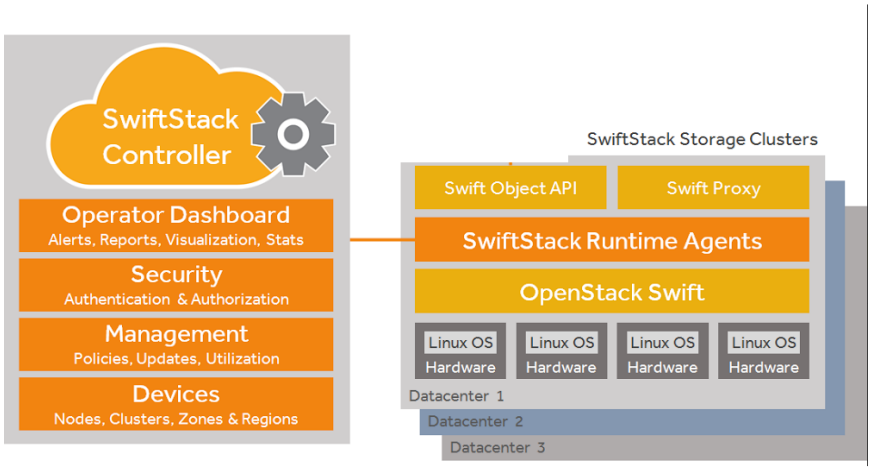
Gateway Hardware

As a SwiftStack component, a Gateway shares the same cost-effective and scalable hardware aspects as the SwiftStack Nodes. This means the Gateway is designed to run on low-cost, industry-standard commodity hardware and installs on standard Linux server distributions. There are no dedicated vendor or hardware specifications that must be used. Instead organizations can determine and make use of the best mix of price, performance and availability.

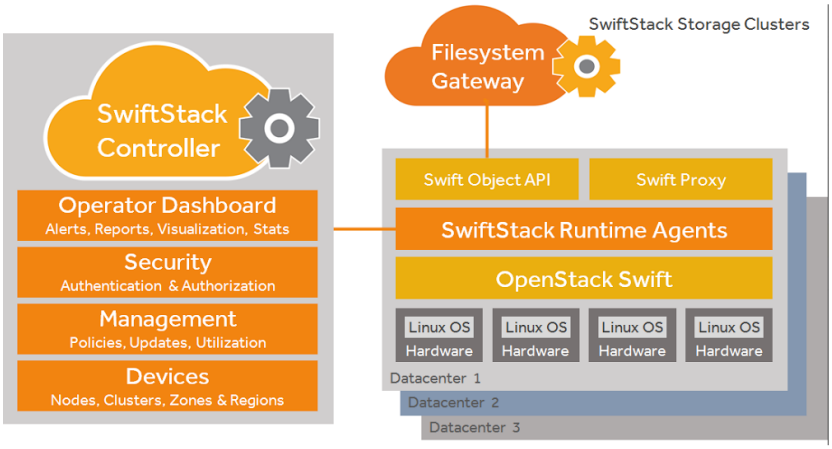
The basic hardware profile for a 1U gateway would include operating system (OS) disks, cache disk and Network Interface Cards (NICs). Commonly deployed on RAID1 to prevent single disk failure, the recommended server OS is CentOS or Ubuntu. While the cache disk size depends on the amount of data going through the Gateway, a suggested minimum would be 1TB. The NICs would be used for the user-facing and cluster-facing network segments. A 1G user-facing NIC and a 10G cluster-facing NIC are a good starting point, however the best configuration will depend on an organization's network configuration. Additional hardware will be needed depending on the use case and functionality that will be needed from the Gateway.

Architecture

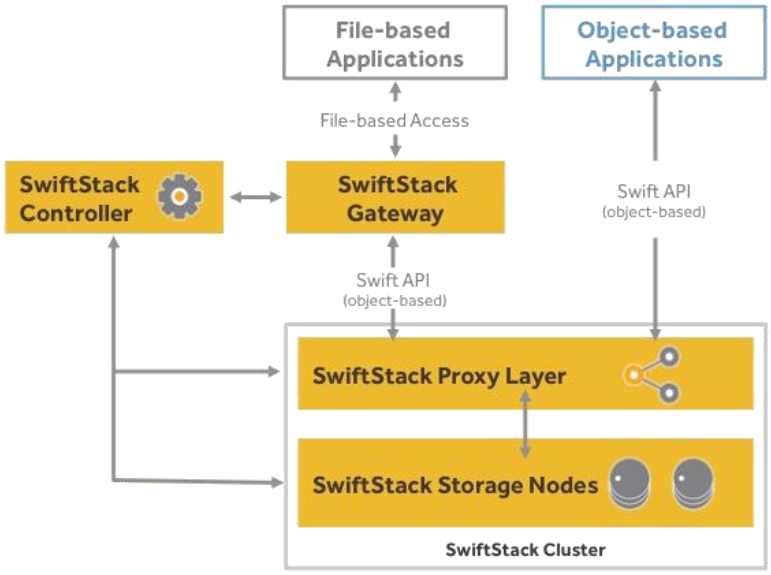
A SwiftStack Cluster is a collection of SwiftStack Nodes that are ingested, configured and monitored using the SwiftStack Controller. The Controller provides an easy, single pane of glass administration for Clusters including authentication and authorization integrations, middleware, monitoring, and alerting for Enterprise environments.



When the Gateway is added to a deployment, its management and monitoring is handled by the SwiftStack Controller. This management includes the configuration of performance options as well as external auth systems such as LDAP/AD. As is the case with SwiftStack Nodes, this management channel for the Gateway is independent of the data path.

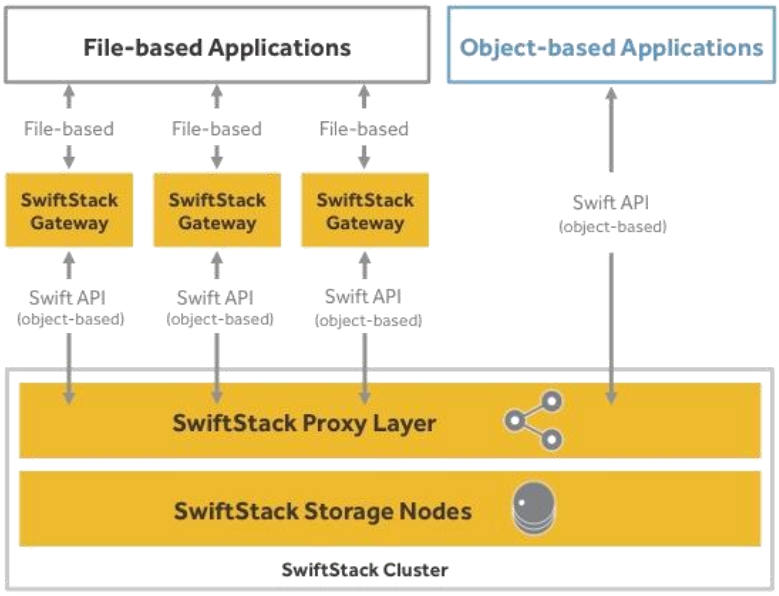


The Gateway provides file-based applications with access to object-based data that is stored in the SwiftStack Cluster. The Gateway handles access requests by checking for a recent copy of the data in its local cache. If the data is not there, the Gateway will retrieve the data from the Cluster. The data is exposed to the file-based application on the Gateway's locally configured filesystem mount point. Using a time-based criterion, the Gateway will identify cached data no longer being written to by the application and move that data back into the Cluster.



Scale Out

As demand increases on the Gateway the most cost-effective and targeted way to meet that need could be to scale out by ingesting additional Gateways and linking them to the Cluster. A multiple gateway solution distributes the demand and ensures there is no single point of failure for the system. Implementing additional Gateways is done on the SwiftStack Controller with no downtime or decrease in performance.



Operational Management

GUI

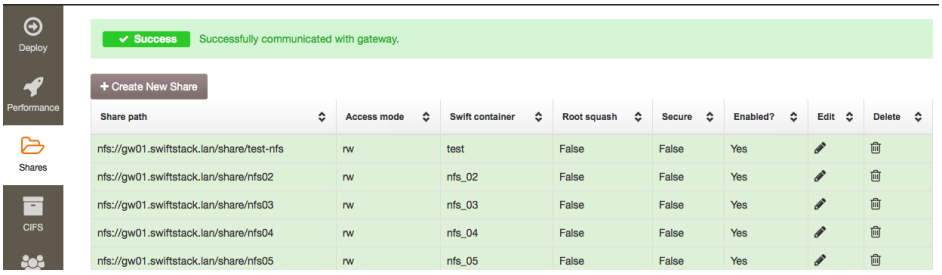
Management of the Gateway is primarily done using the SwiftStack Controller's GUI. The Controller interface is accessible via any web browser. It provides a single place to view and manage the Cluster and the Gateway.

Functionality

In order for the Gateway to provide access to any object in Swift storage as a file, first the correct protocol mount points must be mapped to containers. Once correctly configured and mapped, file-based applications will then be able to mount the desired filesystem and begin accessing their files, which are stored as objects.

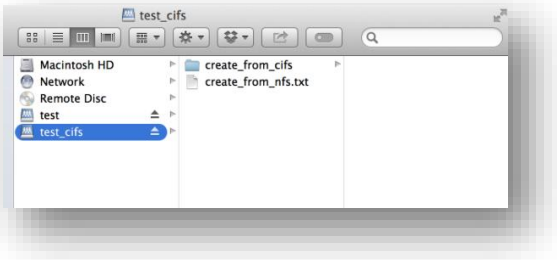
Mapping

A key function of the Gateway is to map filesystem mount points to the containers. Each mapping, called a share, is done using the SwiftStack Controller GUI. Shares are created, configured and edited on the Gateway page of the SwiftStack Controller. Each share maps one filesystem mount point to one container. As a result larger deployments may have hundreds of shares created. Currently the NFS (v3) and SMB (v2.1) protocols are supported, so all shares will either be NFS or SMB/CIFS.

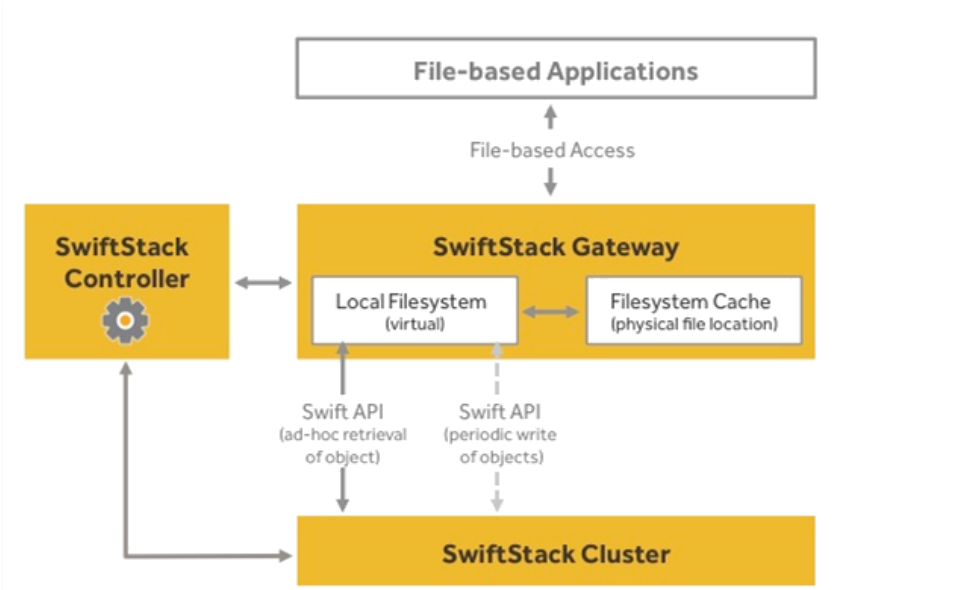


Mounting

Once the shares are created and deployed on the Gateway, user data can be accessed through them by mounting the shares. Shares are mounted on the command line using the 'mount' command or with a client program on a user's computer. Once mounted, the filesystem access point will display the data as files, which can be added, modified or deleted.



Behind the scenes these file-based requests are handled in the Gateway's local filesystem. When a request is made against a file, the Gateway checks for that file in its cache. If the file is not there the Gateway uses an internal reference to determine the file's storage location and retrieves the object using a Swift API GET request to the Cluster. The retrieved file is then available to be modified by the client. All modifications to the file take place in the filesystem with the file physically saved in the cache. Once the "time since the last write", a configurable option, for a file is reached the modified file is written back to the cluster.



Summary

The Gateway exposes object-stored data as SMB/CIFS or NFS files to file-based applications creating a unified storage system capable of active archiving, disaster recovery, content distribution and management, and migration services. As a component of a SwiftStack deployment, the Gateway has the advantages of being built on commodity hardware and managed through the SwiftStack Controller's web interface. This interface provides a centralized location to monitor the health and optimize the performance of the whole deployment. The Controller's web interface is also used to configure the Gateway with the desired filesystem mount points and enterprise integrations, including AD/LDAP. Once deployed the Gateway bridges the two data types, allowing data to be stored and accessed as an object while also being accessed as a file.

Want more information?

For more information on OpenStack Swift, visit swiftstack.com/openstack-swift.

For more information on SwiftStack's software-defined storage platform, visit www.swiftstack.com.