PURESTORAGE®

splunk> turn data into doing™

TECHNICAL WHITE PAPER

Accelerating Splunk at Enterprise Scale

A framework for accelerating Splunk with Pure Storage® all-flash storage systems.

Contents

Introduction **4**

Goals **4**

Audience **4**

Executive Summary of Benefits **5**

Accelerate Splunk: Gain Faster Time to Insights 5

Optimize: Consolidate Infrastructure and Scale Efficiently 5

Simplicity: Reduced Overhead for Splunk Administrators 6

Technology Overview **6**

FlashArray//X 6

FlashBlade 7

Splunk Enterprise 8

CentOS Linux 8

Challenges for Enterprises **8**

Complexity Grows with Scale on Attached Storage 8

Poor or Inconsistent Search Performance 9

The Solution **10**

Enterprise Storage 10

Scalable All-Flash Storage Solution 11

Reference Architecture Design **13**

Design Topology 13

Physical Topology 15

Logical Topology 15

Design Considerations 16

Solution Validation and Testing **19**

Operational Efficiency 19

Data Ingestion 24

Search Behavior 28

Best Practices **33**

Pure FlashArray 33

Linux 33

Pure FlashBlade 33

Linux Mount options 34

Splunk Configuration 34

Conclusion **35**

Splunk Documentation **35**

Appendicies 36
Appendix A: Splunk Enterprise Components 36
Appendix B: Installing Cluster Shell Utility 37
Appendix C: Useful Splunk Searches 39
Appendix C: Useful Splunk Searches 40

Introduction

Pure Storage®, a pioneer of the all-flash array industry, has helped numerous organizations reduce the complexity of their storage management, making their IT more agile and efficient. Splunk® Enterprise is the industry-leading platform for machine data and helps companies solve their complex data management and network security challenges.

IBM says it takes an average of 280 days to detect a data breach, costing an average of $8.4M in damages. Contrast that with the 72 hours that GDPR Article 33 provides to notify authorities of a security breach and you can understand why the job of a security analyst is fraught with challenges. Furthermore, with data growing exponentially security teams need a fast and easy way to parse more data with fewer resources. The traditional Splunk Enterprise indexer cluster deployment model, with direct attached storage, provides high data availability and fidelity but presents significant challenges as data volumes grow.

Splunk customers commonly struggle with the following:

- Inconsistent query performance when searches go after historic data, impacting to user productivity.
- Increased complexity in managing a large number of indexers as the storage needs grow.
- High administrative overhead in keeping the application up and up-to-date, leading to high TCO and low business value.

Cloud options offer an appealing initial value proposition of outsourced infrastructure and infinite scale, but often come with a lack of control and uncontrolled costs over time.

Goals

This white paper explains the benefits and best practices of deploying Splunk Enterprise on Pure Storage's data-focused architectures with FlashArray™ and FlashBlade®. It reports the results of tests of Splunk Enterprise on Pure Storage all-flash arrays. This paper also covers best practices for Splunk deployments on Pure Storage products.

Audience

This document is for system administrators, storage administrators, IT managers, system architects, sales engineers, field consultants, professional services, and partners who are looking to design and deploy Splunk Enterprise on a Pure Storage all-flash storage platform. A working knowledge of Splunk, Linux®, server, storage, and networks is assumed but is not a prerequisite for understanding this document.

Executive Summary of Benefits

This white paper highlights the benefits of Splunk Enterprise on Pure all-flash storage systems (FlashArray and FlashBlade). In this joint solution, Pure FlashArray is used as consolidated block storage for the Splunk hot/warm tier and Pure's fast file and object store FlashBlade for the Splunk cold tier. The paper includes realistic examples of the benefits of the integrated solution, and its capabilities along with best practices for deployment. This white paper demonstrates how running Splunk on Pure Storage on-premises offers an appealing long-term value proposition with the following benefits:

Accelerate Splunk: Gain Faster Time to Insights

Consistent search and indexer performance is essential to keep Splunk users generating business value, preventing security breaches, and helping IT systems to be scalable.

- Users can feel exasperated while searching for critical data on cold tier in the classical architecture or warm-remote storage in the SmartStore architecture. This could be due to slow IO performance while querying cold storage directly or excessive purging and rebuilding of cache in Splunk SmartStore scenarios.
- The Splunk Enterprise on Pure FlashArray and FlashBlade solution provides your users with a consistent search experience across tiers with an all-flash storage system for hot, warm, and cold data while delivering the cost advantages of a consolidated storage solution.
- We tested 120 concurrent sparse and rare searches across hot, warm, and cold tiers. With searches that go after all tiers of data, time-difference between searches on hot/warm data and cold tiers are on average within a few seconds of each other i.e. searches on cold data were almost as fast as searches on hot and warm data.

Optimize: Consolidate Infrastructure and Scale Efficiently

Traditional Splunk deployment with direct-attached storage (DAS) requires the addition of indexers every time more capacity is needed leading to an explosion in complexity at scale. More indexers mean more patching, rebalancing, upgrading, and errors.

- With Pure's all-flash block storage replacing your DAS for hot/warm data, you can now consolidate storage across indexers. This solution lets you scale your storage to multi-petabyte capacity independent of compute reducing complexity and the number of indexers you need. When combined with Pure's multidimensional scale-out storage Flash Blade for cold data, you can scale capacity across tiers without disruption or complexity.
- Splunk recommends an ingest rate of 300GB/day per indexer for Splunk Enterprise . In our tests, we could ingest up to 2.45TB/per day per indexer (40 cores in each server) without CPU usage exceeding 25%. (Note: Results vary with servers, data types, and search volume.)
- Consolidate storage across your entire data pipeline, including other analytics apps such as Kafka, Spark, Vertica, etc. Thin provisioned storage lets you reduce and consolidate storage requirements by sharing storage across servers and platforms.
- Get additional data reduction on top of Splunk compression. The Pure solution can conservatively provide additional data reduction of 1:35:1. (Note: results vary with the cardinality of data.)

Simplicity: Reduced Overhead for Splunk Administrators

Splunk administrators need to focus on value-generating activities such as bringing in new data sources; cleaning, formatting, and enriching them; and building new capabilities for the Splunk User community. Any time spent on operation comes at the expense of value generation. With the Pure solution, administrators can:

- Reduce dramatically the time taken to rebalance data during or after indexer adds, upgrades, or restarts. In our testing, Pure reduced the time taken to add storage from 11 hours and 45 minutes to just 2 seconds and avoided impact to search or indexer performance during capacity upgrade.
- Get built-in data protection and reduced administrative overhead with built-in RAID, erasure encoding, and always-on encryption.
- With Pure as-a-Service™ forget about sizing your storage correctly and pay only for what you use. Help save the planet with Splunk, by reducing power consumption compared to DAS architectures.
- Never worry about upgrading their storage with Pure Evergreen Storage™.

Technology Overview

FlashArray//X

Pure Storage FlashArray//X, the world's first all-flash end-to-end NVMe and NVMe-oF array, now optionally includes a storage-class memory boost to address the most demanding enterprise applications' performance requirements. FlashArray//X delivers breakthroughs in performance, simplicity, and consolidation. It's intended for everything from departmental to large-scale enterprise shared-storage deployments, high performance, and mission-critical applications. Maximize results and flexibility for enterprise and cloud-native, web-scale applications, both on-premises and easily connected to the public cloud. Pure's Evergreen™ model means performance, capacity, and features improve over time without disruption.

- **Accelerate mission-critical applications:** With latency as low as 150 µs, the all-NVME architecture of FlashArray//X with plug-and-play storage-class memory brings new levels of performance and extremely low latency to mission-critical business applications and databases: think faster transactions and decisions and more immersive customer experiences.
- **Hyper-consolidate your cloud:** NVMe enables the unprecedented performance density required for tier 1, mixed-workload consolidation in a private cloud. FlashArray//X currently offers ultra-dense 18.3TB DirectFlash® modules. And Purity's always-on QoS means you can consolidate radically diverse applications without fear of bandwidth or I/O contention.
- **Unify current and future applications:** Organizations have evolved to run a mix of classic business apps with new, modern web-scale apps. This mix previously required radically different architectures. With FlashArray//X, end-to-end NVMe and available NVMe-oF, everything can run on a single shared architecture with the potential to unite storage-area networks (SANs) and DAS. This gives you the performance of DAS while enabling the efficiency, reliability, and simplicity of modern shared storage.
- **Simple cloud-based management:** Pure1® provides simple cloud-based management and effortless predictive support with full-stack analytics and the AI-driven power of Pure1 Meta®. Pure1 provides a snapshot catalog of all your backups in one place, whether the target is another FlashArray, FlashBlade, NFS target, or public cloud like Amazon S3.
- **Evergreen Storage:** FlashArray operates like SaaS and the cloud. Deploy it once and enjoy a subscription to continuous innovation as you expand and improve performance, capacity, density, and/or features for 10 years or more – all without

downtime, performance impact, or data migrations. Pure has engineered compatibility for future technologies directly into the product via the modular, stateless architecture of FlashArray. The Right Size Guarantee™ ensures that you start knowing you'll have the effective capacity you need. The Capacity Consolidation program keeps your storage modern and dense as you expand. With Evergreen Storage, you don't have to re-buy terabytes you already own. Keep your storage evergreen, modern, and dense. And always meet your business needs. Pure uniquely offers all of our core solutions either as products (CAPEX) or as services (OPEX) via our Pure as-a-Service portfolio.

FlashBlade

Pure Storage developed FlashBlade architecture to meet the storage needs of data-driven businesses. FlashBlade is an all-flash system, primarily optimized for storing and processing unstructured data. A FlashBlade system can simultaneously host multiple file systems and multi-tenant object store for thousands of clients.

A FlashBlade system's ability to scale performance and capacity is based on five key innovations:

- **High-performance storage device:** FlashBlade maximizes the advantages of an all-flash architecture by storing data in storage units and ditching crippling, high-latency storage media such as traditional spinning disks and conventional solid-state drives. The integration of scalable NVRAM into each storage unit helps scale performance and capacity proportionally when new blades are added to a system.
- **Unified network:** A FlashBlade system consolidates high communication traffic between clients and internal administrative hosts into a single, reliable high-performing network that supports both IPv4 and IPv6 client access over Ethernet links up to 100GB/s.
- **Purity//FB storage operating system:** With Purity//FB symmetrical operating system running on FlashBlade's fabric modules, Purity//FB minimizes workload balancing problems by distributing all client operation requests evenly among the blades on FlashBlade.
- **Common media architectural design for files and objects:** FlashBlade's single underlying media architecture supports concurrent access to files via a variety of protocols such as NFSv3, NFS over HTTP, and SMB (with Samba-level functionality) and objects via Amazon S3 across the entire FlashBlade configuration.
- **Simple usability:** Purity//FB on FlashBlade alleviates system management headaches as it simplifies storage operations by performing routine administrative tasks autonomously. With a robust operating system, FlashBlade is capable of self-tuning and providing system alerts when components fail.

A full FlashBlade system configuration consists of up to five self-contained, rack-mounted chassis interconnected by high-speed links to two external fabric modules (XFM). At the rear of each chassis, are two on-board fabric modules for interconnecting the blades, other chassis, and clients using TCP/IP over high-speed Ethernet. Both fabric modules are interconnected, and each contains a control processor and Ethernet switch ASIC. For reliability, each chassis is equipped with redundant power supplies and cooling fans.

The front of each chassis holds up to 15 blades for processing data operations and storage. Each blade assembly is a self-contained compute module equipped with processors, communication interfaces, and either 17TB or 52TB of flash memory for persistent data storage.

The current FlashBlade system can support over 1.5 million NFSv3 getattrs per second, or >17 GiB/s of 512KiB reads or >8 GiB/s of 512KiB overwrites on a 3:1 compressible dataset in a single 4U chassis with 15 blades and can scale both compute and performance up to a 5 x 4U chassis with 75 blades.

Splunk Enterprise

Splunk Enterprise makes it simple to collect, analyze and act upon the value of the big data generated by your technology infrastructure, security systems, and business applications, giving you the insights to drive operational performance and business results.

Splunk Enterprise monitors and analyzes machine data from any source to deliver operational intelligence to optimize your IT, security, and business performance. With intuitive analysis features, machine learning, packaged applications, and open APIs, Splunk Enterprise is a flexible platform that scales from focused use cases to an enterprise-wide analytics backbone. Splunk Enterprise provides an end-to-end, real-time solution for machine data, delivering the following core capabilities:

- Universal collection and indexing of machine data, from virtually any source.
- Powerful search processing language (SPL) to search and analyze real-time and historical data.
- Apps provide solutions for security, IT Ops, business analysis, and more.
- Real-time monitoring for patterns and thresholds; real-time alerts when specific conditions arise.
- Powerful reporting and analysis.
- Custom dashboards and views for different roles.
- Resilience and horizontal scalability.
- Granular role-based security and access controls.
- Support for multi-tenancy and flexible, distributed deployments on-premises or in the cloud.
- Robust, flexible platform for big data apps.

CentOS Linux

CentOS version 7.5, a Linux distribution derived from Red Hat Enterprise Linux (RHEL) sources, provides free, enterprise-class computing platform functionalities including the support for Linux Containers. Splunk supports Splunk Enterprise on Linux, Windows, and MacOS. It supports Linux 3.x and 4.x kernel versions for Splunk Enterprise and offers RPM or DEB packages or a tar file depending on the version of the Linux your hosts run. Since a majority of the Splunk implementations have been on the Linux operating system, we opted to use CentOS Linux for our tests.

Challenges for Enterprises

Complexity Grows with Scale on Attached Storage

The Splunk Enterprise indexer cluster deployment model (also known as the distributed scale-out model) provides high data availability and fidelity but also presents significant cost challenges. In this deployment, the Splunk Enterprise indexers are configured to replicate each other's data, thus preventing data loss and facilitating searches in case of a node failure. This model is well suited to earlier Big Data technologies, such as Hadoop, which relied on close server-storage proximity to achieve high performance.

In a distributed scale-out model, every Splunk indexer is configured—predominantly through DAS to have similarly sized storage for hot/warm and cold tiers. This model worked well in the past to process the necessary volumes of data. However, as data grows, storage and compute requirements do not scale linearly. Adding a node of the same type with compute and storage to address the storage requirement is not only suboptimal but also cost-prohibitive.

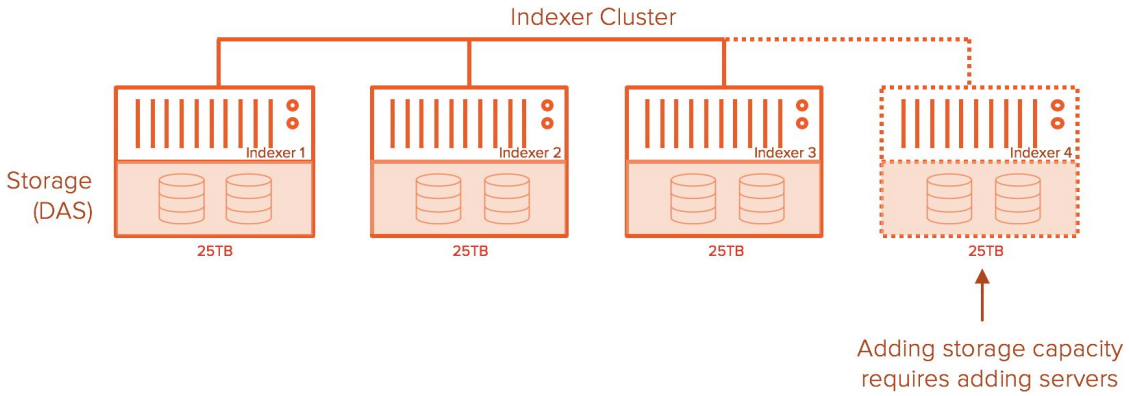

Figure 1: Splunk indexer configuration.

Poor or Inconsistent Search Performance

A key requirement of a distributed scale-out model is the copy or replica of the data on the additional nodes based on the replication factor. An indexer cluster with a replication factor (RF) of 2 and a search factor (SF) of 2 requires the raw data and index data to be replicated to an additional indexer node on top of the indexer node where it got ingested. Hence the storage requirement spikes up considerably in an indexer cluster environment based on the RF and SF. Splunk offers the TSIDX reduction feature to save storage but at the cost of search performance.

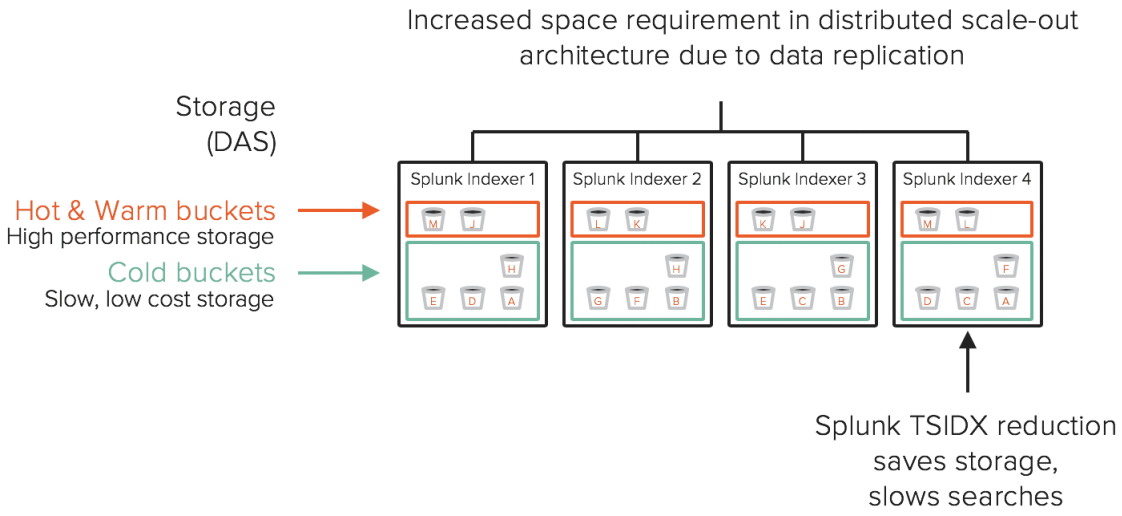

Figure 2: Indexer space requirement in a distributed scale-out architecture.

The exponential growth of data along with the operational and/or compliance requirements to retain that data for a longer time at scale poses significant storage and performance challenges. The high growth storage requirement was handled by tiering the data where recently ingested data or hot data were hosted on faster mediums like SSD drives or PCIe Flash cards while the older data or cold data were stored on low-end HDD drives. This saved cost to store the older or cold data on cheaper disk storage but comes at a cost of slow searches.

With the advent of data analytics, customers are not relying only on the recent data but also trying to get more meaningful insights from all their data. In Splunk, this means the searches are now not limited to the recent data on the hot/warm tiers but also on the cold tier. Big data is only as useful as its rate of analysis and quicker analysis requires faster access to data. Hence in Splunk accessing cold tier data faster requires faster, better storage like all-flash storage as the traditional cheap-and-deep disk storage is inadequate.

The Solution

The solution presented here is comprised of Splunk Enterprise with hot/warm tier on Pure FlashArray and the cold tier on Pure FlashBlade.

- FlashArray//X is the world's first enterprise-class, all-NVMe & NVMe-oF flash storage array. It's a new class of storage – *shared accelerated storage*, a term coined by Gartner, that delivers breakthroughs in performance, simplicity, and consolidation.
- FlashBlade is a ground-breaking scale-out all-flash file and object storage system from Pure Storage architected to consolidate complete data silos while accelerating real-time insights from machine data using applications such as Splunk.

Enterprise Storage

Pure Storage recommends disaggregating storage from the server irrespective of the application to overcome the various inefficiencies exposed by the direct-attached storage. Disaggregation offers efficient utilization of compute and storage while enabling independent scaling of compute and storage to meet the business needs. This means storage space can be added independently by adding storage shelves into the FlashArray or adding blades to the FlashBlade or additional FlashBlade chassis as needed rather than adding more indexer nodes as in the traditional DAS model. This can result in a reduction of indexer nodes which can help reduce the total cost of ownership (TCO) and complexity associated with managing a large number of nodes.

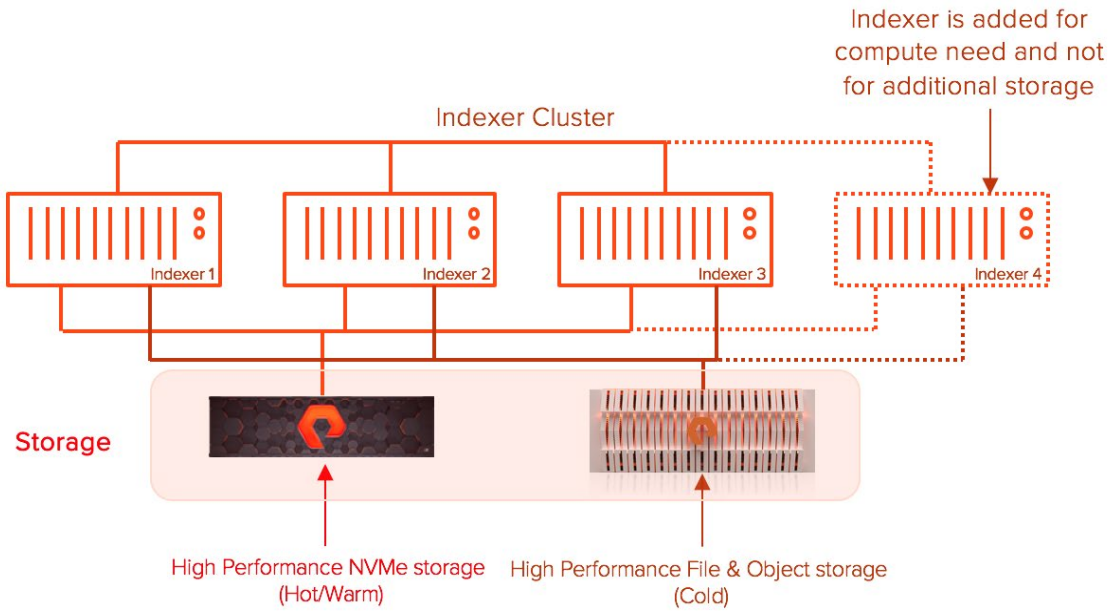

Figure 3: Disaggregating storage.

Scalable All-Flash Storage Solution

Both FlashArray and FlashBlade are thin-provisioned storage systems that offer data services like compression to reduce the storage requirements with the indexer cluster deployment of Splunk Enterprise. As the time series index files (tsidx) are ASCII type in nature, they benefit from the compression at the storage level.

FlashArray offers deduplication service which deduplicates the compressed raw data to a single copy on the storage and helps further reduce the storage requirements. Contrary to the belief that deduplication to a single copy becomes the single point of failure for the application, FlashArray uses RAID-HA, data protection mechanism based on dual parity protection, and dual-controller architecture that eliminates the single point of failure.

The traditional direct-attached storage doesn't offer any data services as it is not only an overhead to implement but requires additional CPU cycles from the host which means the indexer nodes would have fewer CPU resources available to support Splunk activities like ingest and search.

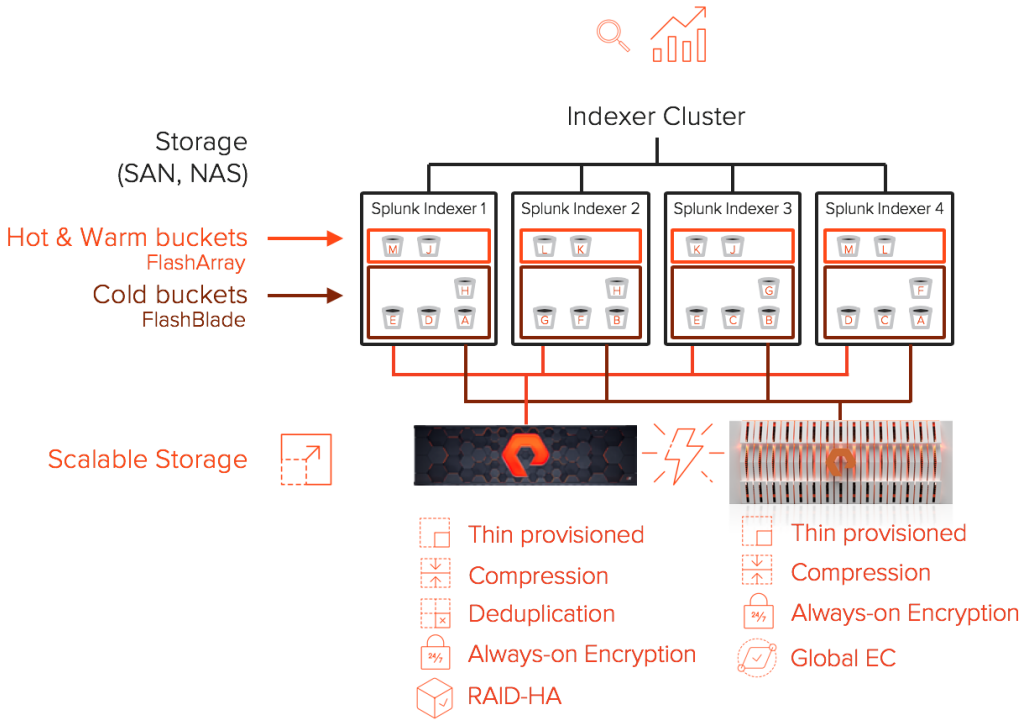

Figure 4: Splunk index clusters with hot, cold, and warm buckets.

The added benefit of enterprise storage like FlashArray and FlashBlade is the in-built data protection through software RAID and erasure encoding that reduces the operational overhead of the system administrators in manually deciding the type of RAID and creating them every time an indexer is added just for added storage. On top of the data protection, both FlashArray and FlashBlade provide always-on encryption services that keep the data encrypted at rest.

The all-flash technology on both FlashArray and FlashBlade offers better performance for the Splunk functions, data ingest, and searches in comparison to that on DAS. Thanks to FlashBlade's technology, the performance characteristics of the cold-tier now matches with that of Hot/Warm tier. Customers can search any available data across Hot/Warm and Cold tiers and get similar performance, thus eliminating any tiered performance and further reaffirming the notion of "no cold data in analytics."

Overall, Splunk Enterprise on Pure Storage systems:

- Provides highly scalable storage solutions.
- Provides all-flash performance across all tiers.
- Enables dynamic cluster scaling by adding compute and storage independently.
- Further reduces space usage through compression.
- Protects Splunk data at rest through encryption.
- Reduces indexer servers while supporting similar or higher capacity requirements.

Reference Architecture Design

Design Topology

This section describes the design topology for the Splunk Enterprise on Pure Storage systems that we tested in our lab.

The solution includes three chassis with 20 Intel CPU-based Cisco UCS B-series M4 and M5 blade servers for hosting the Splunk Enterprise configuration. The Splunk configuration consists of eight indexers, three search heads, one cluster master, and eight universal forwarders. Cisco UCS B-series M5 blades were used for the indexers while M4 blades were used for every other component. The solution includes Pure Storage FlashArray //X70 to host the Hot and Warm tiers while the Pure Storage FlashBlade for the cold tier.

FlashArray Configuration

FlashArray hosts the hot and warm data of Splunk indexes. Separate volume for every indexer node is carved out of the FlashArray and attached to the respective indexer node as a block device over iSCSI. The hot and warm data on FlashArray is deduped, compressed, and encrypted.

Note: The FlashArray volume can be connected over iSCSI or FC to the indexer node.

Component	Description
FlashArray	//X70R2
Capacity	38.34TB raw 26.84 TB usable (with no data reduction)
Connectivity	4 x 40GB/s redundant iSCSI 2 x 1GB/s redundant Ethernet (Management port)
Physical	3U
Software	Purity//FA 5.1.2

Table 1: FlashArray components.

FlashBlade Configuration

The FlashBlade hosts Splunk's cold data, compressed and encrypted at the storage level. Separate NFS filesystem for every indexer is created out of the FlashBlade, connected over ethernet and hard mounted to the indexer.

Component	Description
FlashArray	15 x 17TB blades
Capacity	240TB raw 162.46 TB usable (with no data reduction)
Connectivity	4 x 40 GB/s Ethernet (data) 2 x 1 GB/s redundant Ethernet (Management port)
Physical	4U
Software	Purity//FB 2.4.0

Table 2: Flashblade components.

Operating System and Software Configuration

OS and Software	Description
Linux	CentOS 7.5 (64 bit)
Splunk	Splunk 8.0.5
Cisco UCS Manager	3.2 (1d)

Table 3: Operating system and software configuration.

Physical Topology

The solution, Splunk Enterprise on Pure Storage, consists of a combined stack of hardware (compute, storage, network) and software (Splunk Enterprise, CentOS Linux, Cisco UCS Manager).

Component	Description
Indexer	8 Cisco UCS B200-M5 server blades each with: 2x Intel Xeon Gold 6138 @ 2GHz (20 cores) 256GB of memory
Search Head	3 Cisco UCS B200-M4 server blades each with: 2 x Intel Xeon processor E5-2670 v3 CPUs (12 cores) 256GB of memory
Cluster Master	1 Cisco UCS B200-M4 server blade each with: 2 x Intel Xeon processor E5-2609 v4 CPUs (8 cores) 64GB of memory
Forwarders	8 Cisco UCS B200-M4 server blades each with: 2 x Intel Xeon processor E5-2680 v4 CPUs (14 cores) 128 GB of memory
Networking	2 Cisco UCS 6332 UP 16-Port Fabric Interconnects 2 Cisco MDS 9148S fabric 16G FC Switches 2 Cisco Nexus 9372PX Ethernet Switches
Virtual Interface Card	40 Gbps Unified I/O ports on Cisco UCS VIC 1340
Chassis	3 x Cisco UCS Chassis 5108 (6RU each)

Table 4: Topology of Splunk Enterprise on Pure Storage.

Logical Topology

For enterprise-grade deployment of Splunk Enterprise that requires multiple terabytes of data to be ingested daily while supporting numerous concurrent searches, we recommend a distributed deployment of indexer clustering. The configuration that we tested in our lab was:

- Eight Indexers in an indexer cluster setup
- Three search heads in a search head clustering
- One cluster master
- Eight universal forwarders

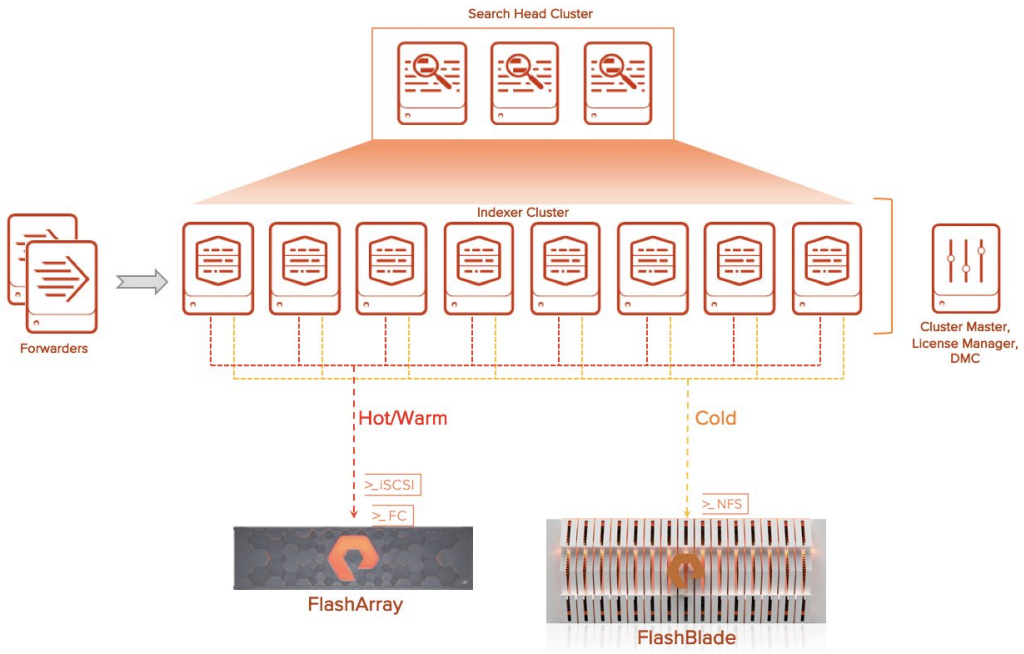

Figure 5: Tested distributed deployment configuration.

Design Considerations

Splunk Enterprise deployments are generally based on the following factors:

- **Daily ingest rate or indexing volume:** Higher index rates require more indexers.
- **Number and type of searches:** Numerous concurrent searches or resource-intensive searches (i.e. dense search) can tax the search heads and indexers.
- **Number of concurrent users:** Numerous users viewing dashboards or running searches would require more search heads, ideally a search head cluster.
- **Data fidelity:** An indexer cluster is required to ensure against data loss.
- **Data availability:** Deploy both an indexer cluster and a search head cluster to get access to the full set of data.
- **Disaster recovery requirements:** A multisite indexer cluster spread between two data centers enables you to maintain identical data sets for quick recovery.

Splunk Enterprise manages the data fidelity, availability, and disaster recovery requirements at its software layer by replicating the index data.

Capacity

Splunk offers indexer clusters to prevent data loss while promoting data availability for searching by index replication where Splunk keeps multiple copies of incoming data. While the index replication provides the benefits of data availability, it comes with the cost of additional storage to hold the replicated data along with a smaller degree of processing load on indexers and additional network traffic to replicate the data between indexers.

There are two key factors (named *replication factor* and *search factor*) that determine the data availability requirement either for fault-tolerance or search:

- Replication Factor (RF)
  - Determines the indexer cluster's failure tolerance.
  - Determines the number of copies of data that the indexer cluster maintains.
  - Cluster can tolerate a failure of (RF -1) peer nodes.
  - Contains the rawdata in compressed format.
  - Exact copy on peer nodes.
- Search Factor (SF)
  - Determines the number of searchable copies of data the indexer cluster maintains.
  - Contains both the rawdata in a compressed format and index files.
  - Logical copy on peer nodes as the index data is generated locally on the peer node based on the rawdata that was replicated.
  - Increased search factor doesn't mean improved search performance rather improved search availability.
  - Recommended and the default value is two.

Storage Sizing Guidelines

To size the storage requirements for Splunk in an indexer cluster deployment model, the following entries are needed. For clarity, the sizing doesn't include the frozen storage calculations.

- Daily Ingestion Rate – Recommended to use the Splunk daily license usage
- Retention period (includes both Hot/Warm and Cold usage)
- RF and SF

At a high level, the storage requirement for Splunk is calculated as follows:

$$Storage\ Required = retention\ period\ *\ daily\ ingest\ data\ *(0.15 * RF + 0.35 * SF)$$

Note: The Splunk compression ratio is generally 50% (made up of 15% of the ingested data towards raw data and 35% of the ingested data towards index metadata) but this is entirely dependent on the type of data. For higher cardinality data, this percentage can go down, resulting in lower compressed data overall or an increase in the storage sizing requirement.

To size the storage requirements for both FlashArray and FlashBlade to support this solution, you need the additional two items:

- Hot/Warm tier retention period (HWR)
- Cold tier retention period (CR)

The storage requirements for Hot/Warm tier on FlashArray is calculated as follows:

$$FA\ Storage\ required = \ daily\ ingest\ data\ * HWR * \ (0.15 + 0.35 * SF)$$

The storage requirement for the Cold tier on FlashBlade is calculated as follows:

$$FB\ Storage\ required = \ daily\ ingest\ data\ * CR * \ (0.15 * RF + 0.35 * SF)$$

> **Note:** These calculations are estimates and do not include the data reduction ratios offered by FA or FB as well as the RAID overhead. Please work with your Pure Storage account team to get the right size requirements for your Splunk environment.

As you can see from the above calculations, the replication factor doesn't make a difference with the Hot/Warm tier calculation on FlashArray because the multiple copies of the rawdata are always deduped at the storage level, rendering extra storage space. If you are worried about the deduped single copy of the raw data being the single point of failure, FlashArray uses RAID-HA data protection that offers recoverability from one or two simultaneous read failures in a group of data blocks.

For example, if you have a daily ingest rate of 4TB per day with Hot/Warm retention of 90 days and Cold retention of 270 days with RF=2 and SF=2, the total storage requirement is:

- Storage required = 360 * 4TB * (0.15 * 2 + 0.35 * 2) = 1,440TB
- Hot/Warm requirement = 90 * 4TB (0.15 * 2 + 0.35 * 2) = 360TB
- Cold requirement = 270 * 4TB (0.15 * 2 + 0.35 * 2) = 1,080TB

The usable FlashArray storage requirement to support Hot/Warm would range between 140 and 180TB based on the compressibility of the index data in addition to the deduplication of the rawdata.

The usable FlashBlade storage requirement to support the Cold tier would range between 600 and 750TB based on the compressibility of the index data. In both cases, the actual raw storage would be different to account for the RAID or EC overhead.

Performance

Planning system resources and bandwidth to enable search and index performance in a distributed indexer cluster environment must factor in the total volume of data being indexed and the number of active concurrent searches (scheduled or other) at any time.

As per Splunk's performance recommendation, an indexer that meets the Splunk's reference hardware specifications can ingest up to 300GB/day while supporting search load in a Splunk Enterprise deployment and 100GB/day in an Enterprise Security deployment. Splunk has introduced two new hardware specifications for better indexing performance and search concurrency over a distributed Splunk Enterprise deployment. The mid-range specification recommends 24 CPU cores at 2GHz or greater with 64GB RAM while the high-performance specification recommends 48 CPU cores at 2GHz or greater and 128GB RAM.

Since Splunk search heads and indexers are CPU-intensive, it is recommended to split the search and indexing functions with distributed searching enabled. This enables search heads to distribute parallelized searches.

For further performance recommendations, please refer to Splunk Enterprise Capacity Planning Manual.

Solution Validation and Testing

The architecture goal of the Splunk Enterprise at scale is to enable fast indexing and improve search responses irrespective of the type of search or the storage tiers while eliminating the challenges with direct-attached storage.

The Splunk solution on Pure Storage systems is validated by testing the below key functions of Splunk Enterprise, namely data ingestion and search and also the operational efficiency of Splunk space management:

- Operational efficiency
- Data ingestion
- Search behavior

Operational Efficiency

With machine data becoming prevalent, the conventional wisdom on space management with Splunk is to anticipate growth all the time. Hence, managing the storage space is often a challenging activity for the Splunk administrator. The operational efficiency test for storage management will showcase the benefits of enterprise storage like Pure FlashArray and FlashBlade in adding storage space to the indexer nodes.

As the storage is disaggregated from the compute in the Splunk Enterprise on Pure Storage solution, operational activities like the addition of storage space to the indexer cluster is quicker and simpler than the traditional approach.

Storage Management Test Overview

The following sections cover how storage space is added in the traditional approach vs. Pure Storage and how the respective tests were performed.

Traditional Approach

In the traditional direct-attached storage model, the indexer nodes are pre-populated with the storage devices like SSD or HDD on the available slots on the server. The best practice for setting up these storage devices is in a RAID format like RAID-1 to avoid a single point of failure but at the cost of double the storage space. If an indexer node runs out of space, the standard practice is to add a new peer node to the cluster and rebalance the cluster to distribute the data evenly across all the nodes.

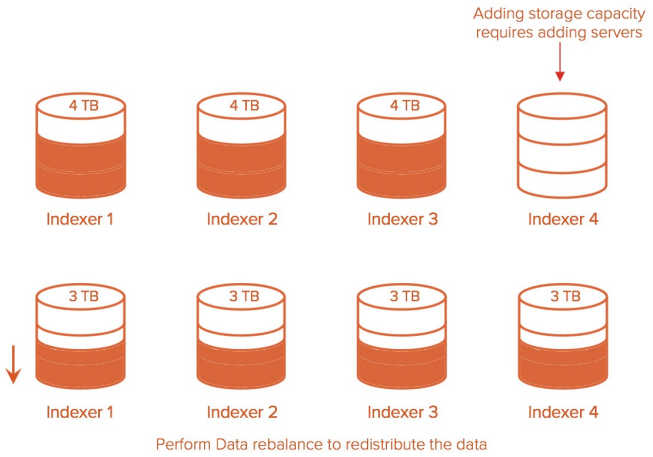

Figure 6: Rebalancing to redistribute data.

It is not only expensive to add a new indexer to support the additional storage space but also time-consuming to perform the data rebalance to distribute the bucket data across all nodes within the cluster. The eight-node indexer cluster had a total space of 60TB across 67,000 buckets. In this test, we added an indexer to add space, performed the data rebalance using the following command, and measured the total time.

```
$SPLUNK_HOME/bin/splunk rebalance cluster-data -action start -auth admin:splunk123
```

Pure Approach

In the disaggregated model, enterprise storage for the whole indexer cluster replaces the individual direct-attached storage medium from the server. As such, the function of adding space is offloaded to the enterprise storage solution.

In the case of the Pure FlashArray, it is easier and simpler to add a new volume for each indexer peer, attach them to the peer node, and add them to the logical volume manager to start using the additional storage. There is no need to add a new server to get additional space.

In the case of the Pure FlashBlade, the addition of storage cannot be any simpler than just to edit the NFS filesystem on the GUI or through CLI and increase the size and the indexer will automatically reflect the new size.

The current filesystems for the Cold tier were set at 10TB. In this test, we increased the space to the cold tier across all eight indexers in the cluster, by updating the FlashBlade filesystems through the following CLI commands.

```
purefs setattr --size 20T splunk-cold-ix1
purefs setattr --size 20T splunk-cold-ix2
purefs setattr --size 20T splunk-cold-ix3
purefs setattr --size 20T splunk-cold-ix4
purefs setattr --size 20T splunk-cold-ix5
purefs setattr --size 20T splunk-cold-ix6
purefs setattr --size 20T splunk-cold-ix7
purefs setattr --size 20T splunk-cold-ix8
```

Operational Efficiency Test Results

Before adding the new indexer node for additional space, the bucket usage across all indexers was captured by selecting "Indexer Clustering" under Settings => Distributed Environment.

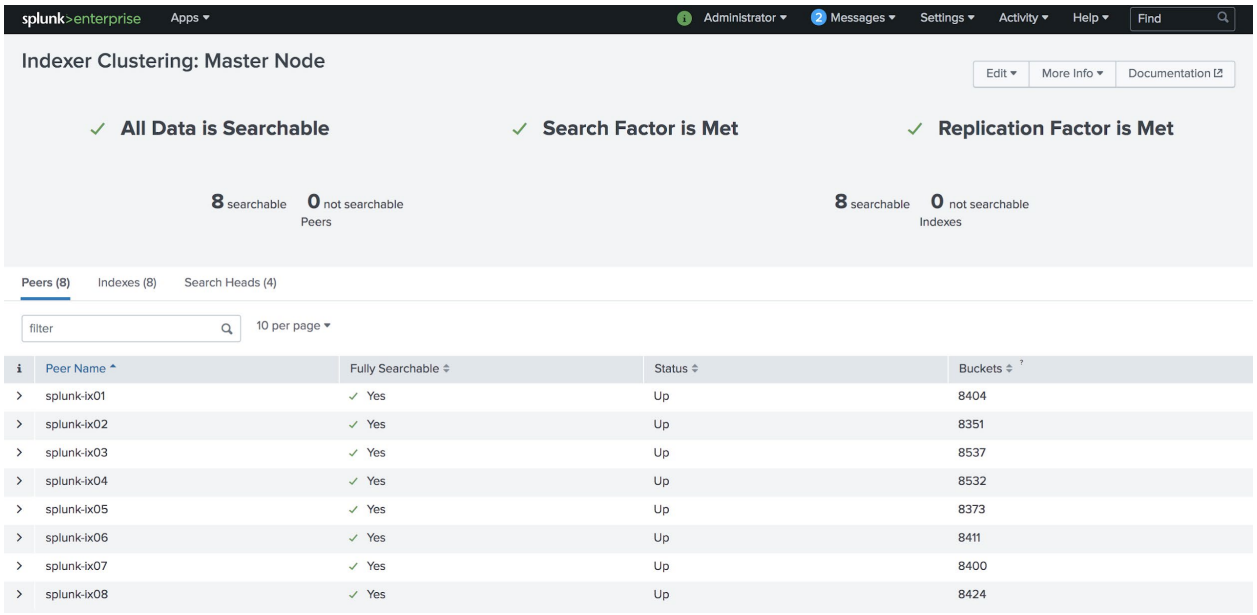

Figure 7: Bucket usage across indexers.

Once the indexer node was added, the data is not rebalanced by default and the new indexer shows bare-minimum buckets.

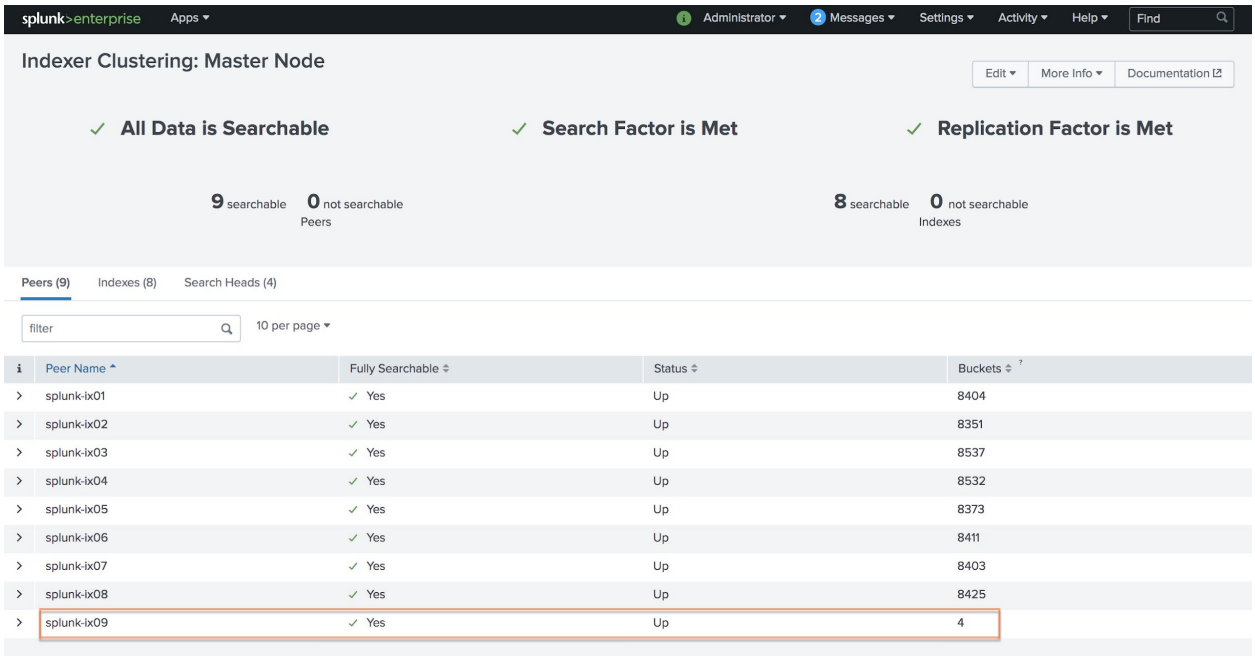

Figure 8: New indexer with minimum buckets.

The data rebalance command was invoked through CLI which took **11 hours and 45 minutes** to complete. At the end of the data rebalance the buckets were uniformly distributed across all the 9 indexers.

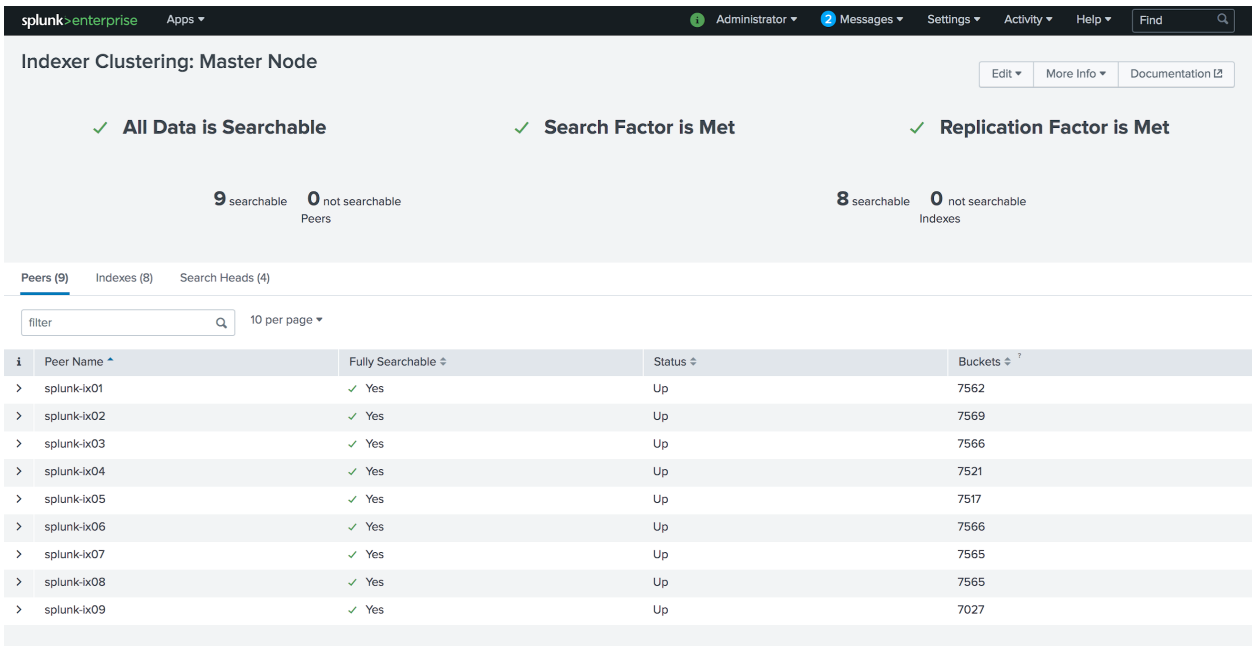

Figure 9: Uniform bucket distribution after data rebalancing.

The bucket fixup details during the data rebalance show the “to_fix_rebalance” activity for the duration. During this process, Splunk replicated the data across the nodes as part of redistribution, made the buckets searchable, and truncated the size of the buckets that were moved.

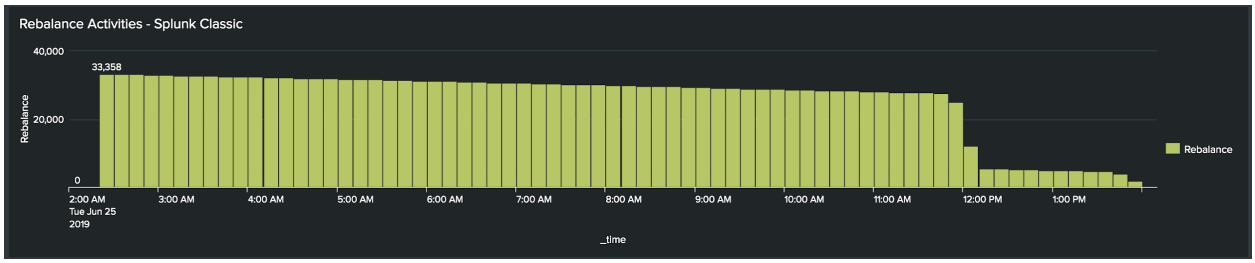

Figure 10: Rebalancing activity over time.

Cold Tier Space increase on FlashBlade

The space usage of the cold tier volume was captured before the space increase. For the sake of brevity output from the first indexer is shown below.

```
[root@splunk-ix01 ~]# df -h /cold
Filesystem                      Size  Used Avail Use% Mounted on
10.21.214.203:/splunk-cold-ix01   10T  6.6T   10T  66% /cold
```

Figure 11: Output from the first indexer.

The CLI to increase the cold tier filesystem space was executed on the FlashBlade which completed within two seconds for all the eight filesystems.

```
pureuser@sn1-fb-d077-am2> purefs setattr --size 20T splunk-cold-ix01
Name             Size  Used     Hard Limit  Created                  Protocols
splunk-cold-ix01  20T    6.6T    False       2019-05-16 18:17:28 PDT  nfs
pureuser@sn1-fb-d077-am2>
```

Figure 12: Completed CLI.

Checking the same NFS filesystem mounted on the first indexer node now shows the increased storage size.

```
[root@splunk-ix01 ~]# df -h /cold
Filesystem                      Size  Used Avail Use% Mounted on
10.21.214.203:/splunk-cold-ix01   20T  6.6T   10T  33% /cold
```

Figure 13: Increased storage size.

Table 5 summarizes the storage space addition test results.

Operational Activity	Traditional Approach	Pure Approach
Storage Space Addition	11hr 45 min	2 sec

Table 5: Traditional vs. Pure storage space test results.

As you can see, adding space in Splunk Enterprise with disaggregated storage is far simpler, quicker, and more importantly, cheaper. While an argument can be made as to why data rebalancing is required, if it is done purely to address the space usage, it can be completely avoided with the disaggregated storage option with Pure Storage products.

Data Ingestion

Overview

During data ingestion, a Splunk Indexer indexes the incoming data by transforming raw data into events based on timestamps and stores them in index files. Splunk stores all its data in buckets on the indexer nodes. Buckets are simply index directories hosted on the indexer nodes. A bucket moves through various stages as it ages via Hot, Warm, Cold, Frozen, and Thawed buckets.

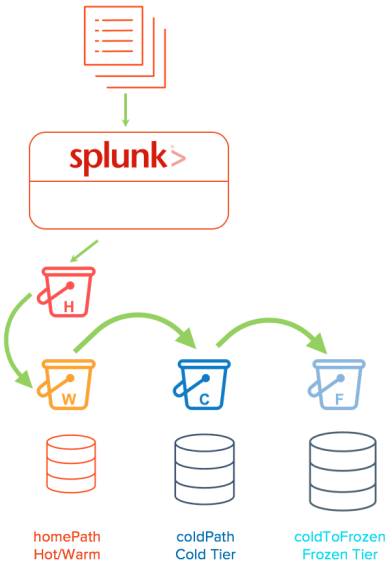

Figure 14: Splunk bucket aging.

When data is indexed, it lands in a hot bucket, which is both searchable and actively being written into. Hot is the only bucket that can be written to. When conditions like a hot bucket reaching its pre-configured size or reaching the maximum number of hot buckets, the hot bucket rolls into a warm bucket. Warm buckets are searchable but cannot be actively written into. When further conditions are met, like reaching the maximum number of warm buckets, the indexer rolls the warm buckets to cold, based on their age. The oldest warm buckets are the first candidates to be rolled to the cold bucket.

In this solution, both the hot and warm buckets are hosted on FlashArray while cold buckets are hosted on FlashBlade. Even though the hot buckets are where the data get ingested, the buckets will always be rolling from hot to warm and warm to cold throughout the day at Enterprise scale, which makes the write performance of a cold tier as important as that of a hot tier.

Ingest Test Overview

We tested ingest 64TB of data into the indexer cluster with eight peer nodes with all indexes configured with both replication factor and search factor set to 2. We measured the time taken to load 64TB of data as well as capture system utilization and other pertinent metrics.

Ingest Test Setup

To scale test the Splunk Enterprise ingestion, we ingested 4 batches of 16TB amounting to 64TB of overall data. The test data we opted to generate was the apache type log of sourcetype access_combined with additional keywords on every event that can be used for sparse and rare searches. The dataset was generated through a custom-developed script using Go programming language. The data generation script was run on eight forwarders which generated 2TB per server totaling 16TB for each batch.

To speed up the data through forwarders, we updated the throughput entry maxKBps in $SPLUNK_HOME/etc/system/local/limits.conf to 0 which means there was no regulation of the data throughput from the forwarder to the indexers. The default throughput of universal forwarders is 256KBps. For testing purposes, larger throughput (e.g. 10240 or 0) might be acceptable; but for standard operating procedures, you might want to set this to a value that reflects your environment so as to not overwhelm your network infrastructure.

The 16TB of data was ingested into four indexes named apache-pure, apache-pure2, apache-pure3, and apache-pure4 sequentially from the eight universal forwarders using the one-shot method.

```
$SPLUNK_HOME/bin/splunk add oneshot <source log file> -index apache-pure2 -sourcetype access_combined -auth admin:splunk123
```

Ingest Test Results

The data ingestion of 64TB from eight universal forwarders into the indexer cluster with eight indexers, took 78 hours 17 minutes at an average rate of 238MBps or 19.62TB/day.

Ingested Data	Elapsed Time	Indexing Rate	Indexing Rate per Indexer	Daily Ingest Rate
64TB	78hr 45 min	238.12MB/s	29.76MB/s	19.62TB/day

Table 6: Ingest test results.

Figure 15 shows the indexing rate of a 16TB batch load to one of the indexes and as you can see the indexers reached the peak indexing rate of 272MB/s and averaged at 238MB/s.

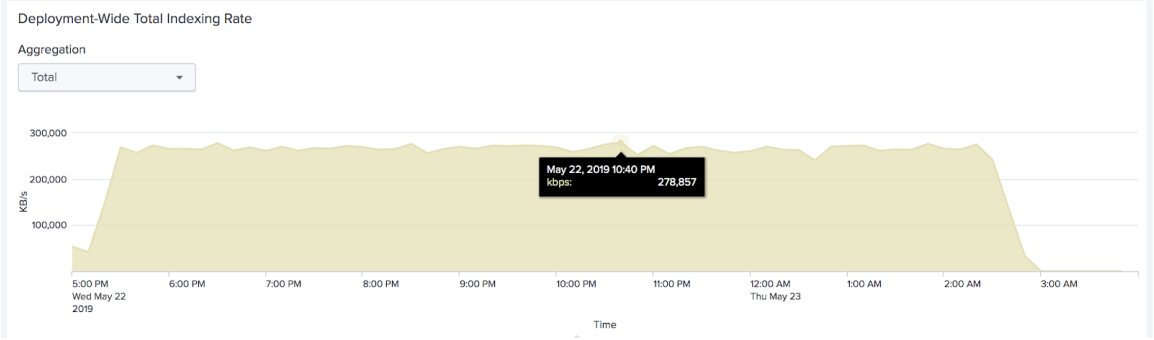

Figure 15: Indexing rate of a 16TB batch load to one of the indexes.

During the ingest, the overall CPU utilization of all the eight indexer nodes was under 25%. The following screenshots show the performance metrics of both FlashArray and FlashBlade during the ingest process.

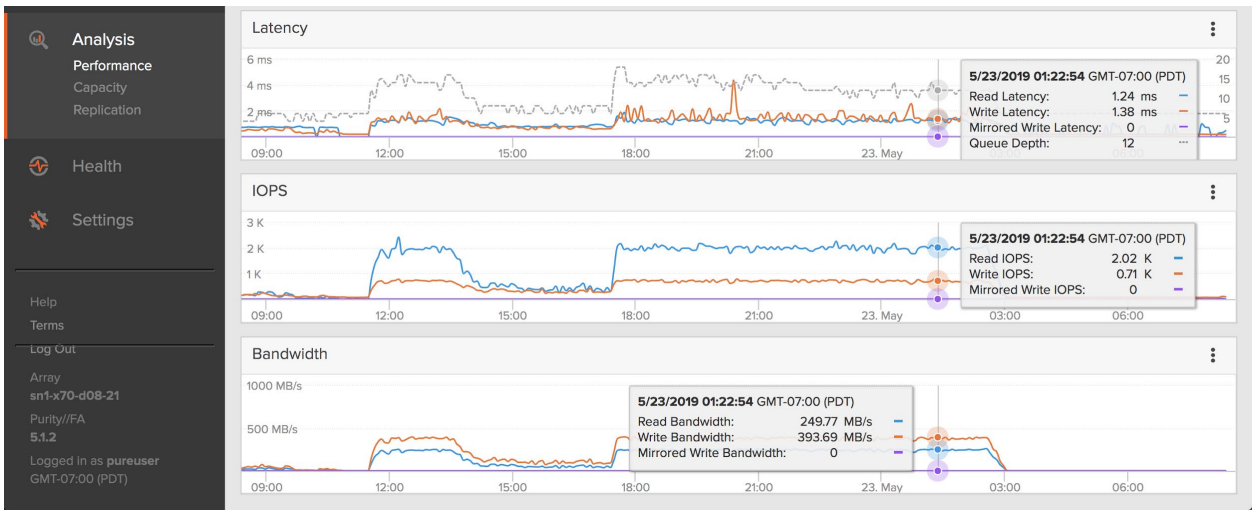

Figure 16: Performance metrics of FlashArray during the ingest process.

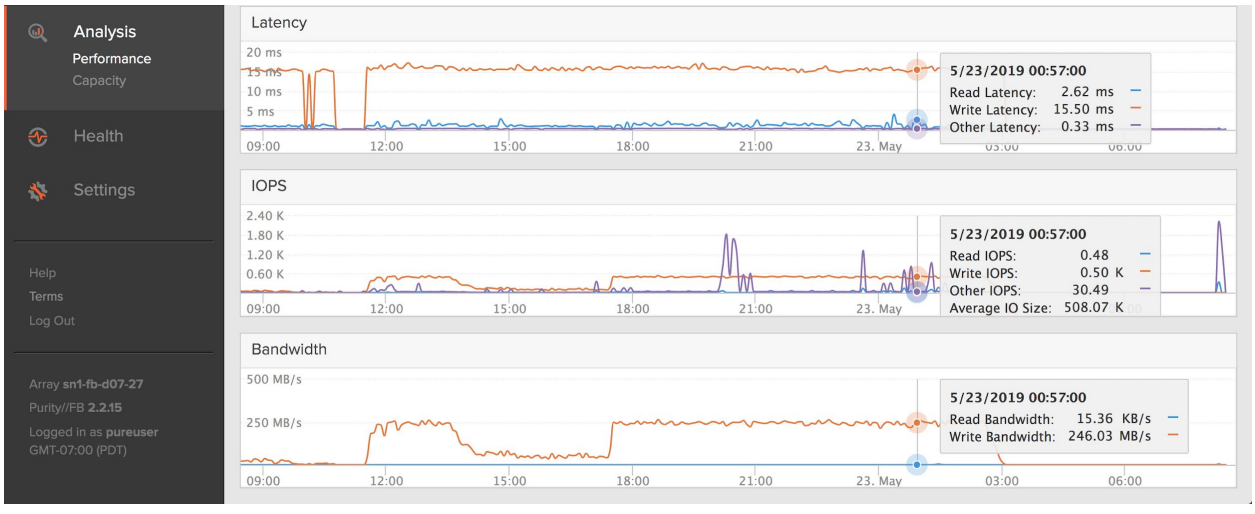

Figure 17: Performance metrics of FlashBlade during the ingest process.

As the buckets were constantly rolled from Hot to Warm to Cold during the large scale of ingest, both FlashArray and FlashBlade reflected the same in terms of the write activity.

Based on the test results, both FlashArray and FlashBlade were never stressed hard during the 64TB of ingest process and both these systems have more processing power left to take up additional workloads.

Space Usage

Splunk compresses the incoming raw data and along with index data generation, offers around 50% data reduction on the ingested data. In the case of indexer clustering, this data reduction would be offset by the RF and SF copies of the rawdata and index data to provide high availability of data. The data reduction can be further impacted by the cardinality of the data.

Hence, the 64TB of ingested data on Splunk Enterprise with RF=2, SF=2 resulted in Splunk consuming 59.84TB of space. This is at 1.06: 1 data reduction at the Splunk level.

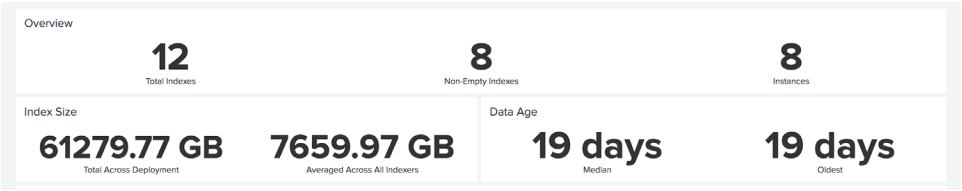

Figure 18: Overview of indexes.

In addition to the data reduction by Splunk, both FlashArray and FlashBlade compress the Splunk data, resulting in a further data reduction which depends on the type of data and cardinality. Based on the ingest data used in this testing, which was of higher cardinality, the data reduction was at 1.35:1 with RF=2 and SF=2.

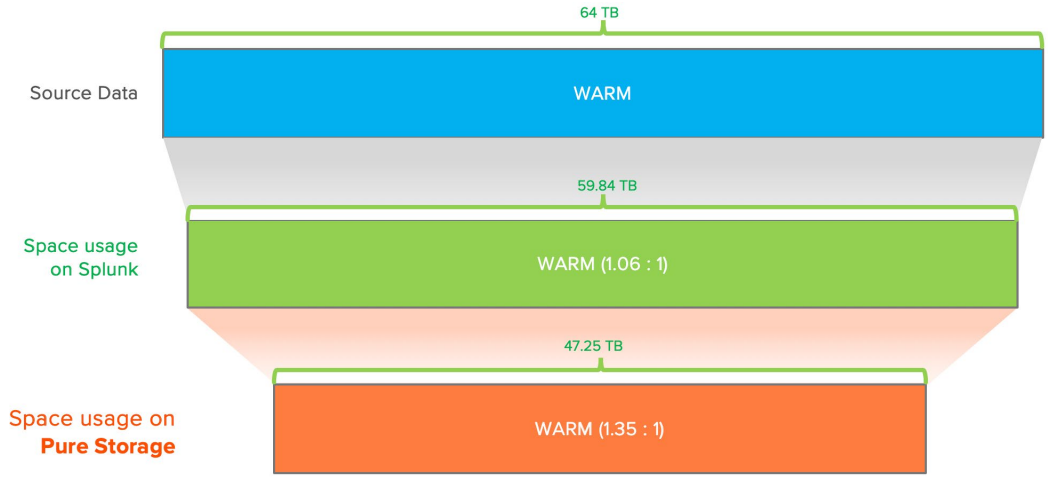

Figure 19: Data reduction using Pure Storage.

Splunk's recommended daily ingestion rate per indexer based on reference hardware with 12 cores at 2GHz is 300GB/day for Splunk Enterprise while supporting some search load. This equates to **2.4TB/day** of ingest as per our eight-indexer node configuration. For enterprise-scale, Splunk recommends high-performance reference hardware with 48 cores, which can yield higher than 300GB/day ingest rate while supporting search load.

In our test, with concurrent search load, the theoretical daily ingest that was achieved with eight indexer nodes under 25% of CPU utilization and with healthy indexing queues is **19.61TB/day**. This is because we selected Cisco UCS M5 blades for indexer nodes which are made up of 2x20 CPU cores (40 cores) in total. This aligns with Splunk's high-performance reference hardware's performance capability and demonstrates the possibility of improving performance on a superior infrastructure.

Search Behavior

The majority of the Splunk customers perform their searches on near-term data which generally resides on the Hot/Warm tier. Hence the Splunk best practice is to place the Hot/Warm data on faster storage to improve search performance.

As more and more Enterprise level customers become interested in gaining more insights into all the data they have and rather than just near-term data, the requirement to search the data across the cold tier has gone up. At the same time, customers don't want to wait for the longer searches across the cold tier to complete. They wanted the searches that go after the long-term data to respond as quickly as it would on the near-term data.

Search Test Overview

We performed the following tests and measured critical metrics like the elapsed time, eliminated bucket info, etc.:

- 120 concurrent searches (spare and rare) – 100% Hot/Warm tier
- 120 concurrent searches (sparse and rare) – 90% Hot/Warm tier, 10% Cold tier

The first search performs all 120 concurrent searches on the Hot/Warm tier while the second test performs 108 searches on the Hot/Warm tier and the remaining 12 searches on the Cold tier simulating 10% searches on the cold tier. In either case, two types of searches, sparse and rare, were performed. The objective of this search test is to showcase the performance of searches across the Hot/Warm tier that are hosted on FlashArray along with searches across cold tier that are hosted on FlashBlade. In a traditional setup, the searches against the Cold tier would be slower as they are generally stored on cheaper disk drives.

Search Test Setup

The 64TB of dataset across 4 indexes (or 16TB per index) were distributed across 16 days at 1TB per day. Each index on an average takes up to 500 buckets per day and the bucket size (maxDataSize) for all indexes is set to **auto**, which is 750MB.

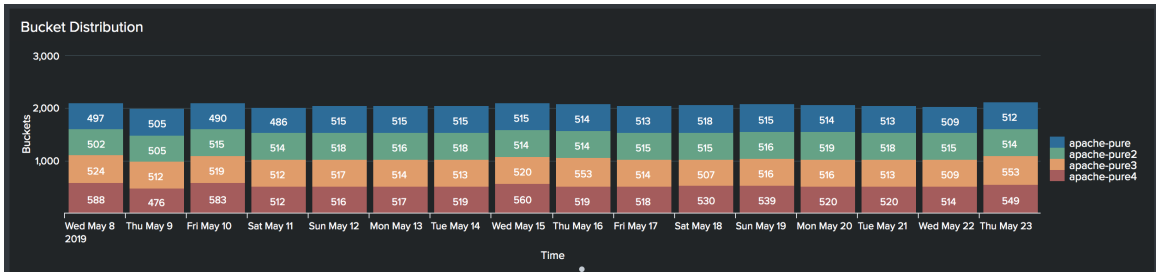

Figure 20: Bucket distribution over time.

To simulate the real-world scenario where predominantly 10% of the total data is on the Hot/Warm tier and the remaining on the Cold tier, our Hot/Warm tier would amount to 6.4TB.

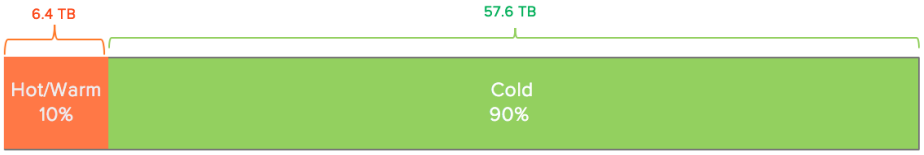

Figure 21: Data distribution in Hot/Warm and Cold tiers.

We used the following calculation to get the right homePath max data size:

```
Hot/Warm size per apache index = 10% * Indexed data / # Indexers / # Apache indexes
         ⇒ 10/100 * 64TB / 8 / 4 = 200 GB per index
```

Hence, we set the homePath.maxDataSizeMB on each apache index to 200GB on the indexes.conf file as given below.

```
[apache-pure]
homePath = volume:hot/$_index_name/db
coldPath = volume:cold/$_index_name/colddb
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
homePath.maxDataSizeMB = 204800
# 200 GB of Hot/Warm
```

Concurrent Searches

The concurrent searches test were performed against two dimensions:

- Search type (sparse and rare)
- Hot/Warm tier (100% and 90%)

In these tests, 120 concurrent searches were run with 40 searches against each search head. The 120 concurrent searches enable us to test 15 searches per peer node (even though there is no guarantee it will be equally spread across all peer nodes). Table 7 illustrates the search type (sparse and rare), per search dataset in terms of size, events per search, and the expected events.

Search Distribution	Search Type	Data set per search	Events per search	Total events per search	Searches on Hot / Warm	Total Hot / Warm Data set	Searches on Cold Tier	Total Cold Data set
100%	Sparse	15GB	9820	98.2M	120	1.8TB	0	-
90%	Sparse	15GB	9020	98.2M	108	1.6TB	12	180GB
100%	Rare	10GB	2	32.7M	120	1.2TB	0	-
90%	Rare	10GB	2	32.7M	108	1.08TB	12	120GB

Table 7: Search type (sparse and rare), per search dataset in terms of size, events per search, and the expected events.

In the case of 100% concurrent searches, all 120 searches go after the data in the Hot/Warm tier. As each sparse search goes after approximately 15GB of Splunk data, running all 120 searches are expected to cover 1.8TB of Splunk dataset; the rare search goes after 1.2TB of Splunk data in the Hot/Warm tier. In the case of concurrent searches at 90% Hot/Warm tier, 90% of the searches (108) go after the data in the Hot/Warm tier and 10% of the searches (12) go after the data that in the Cold tier.

The searches were submitted through curl using the REST endpoints for the searches. The search results were extracted through the same mechanism. Also, the operating system's cache was cleared prior to every search to get meaningful I/O response times.

Search Test Results

The Search Usage Statistics: Deployment under DMC => Search => Activity shows the search activities and the following screenshot shows the 120 concurrent sparse search details under 100% cache-hits.

Search Activity by User (1)

User	Search Count	Search Head Count	Median Runtime	Cumulative Runtime	Last Search
admin	120	3	0.93s	1min 52.41s	07/18/2019 23:16:33 -0700

Click to see a list of search head names and a list of search strings.

Search Activity by Search Head (3)

Search Head	Search Count	User Count	Median Runtime	Cumulative Runtime	Last Search
splunk-sh01	40	1	0.87s	35.73s	07/18/2019 23:16:33 -0700
splunk-sh02	40	1	0.95s	38.55s	07/18/2019 23:16:33 -0700
splunk-sh03	40	1	0.93s	38.13s	07/18/2019 23:16:33 -0700

Click to see a list of users and a list of search strings.

Figure 22: The 120 concurrent sparse search details under 100% cache-hits.

Figure 23 illustrates the search results of the 120 concurrent searches all against the Hot/Warm tier hosted on FlashArray. Each sparse search took 0.93 seconds to return 9820 events out of 98.2 million events while each rare search took 0.28 seconds to return two out of 32.7 million events.

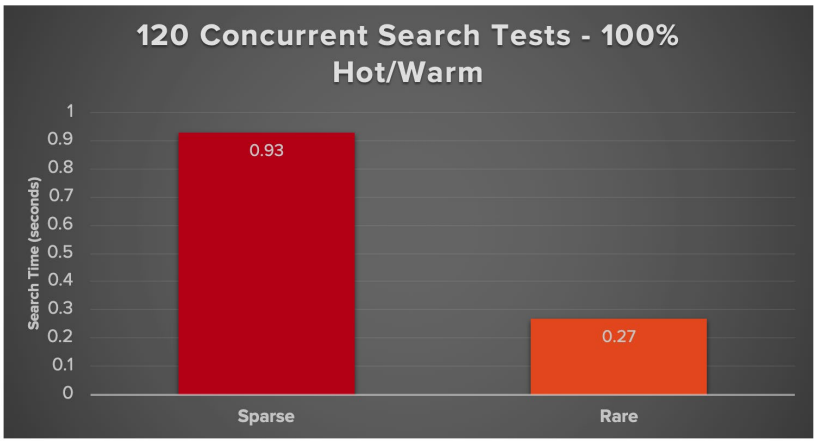

Figure 23: Search results of the 120 concurrent searches all against the Hot/Warm tier hosted on FlashArray.

Figure 24 shows the run time of the sparse and rare searches with 100% Hot/Warm tier as well as both the searches with 90% against Hot/Warm tier and 10% against the Cold tier. This means that out of the 120 searches, 90% of the searches, or 108, went after the data in the Hot/Warm tier, and the remaining 10%, or 12 searches, went after the data on the Cold tier.

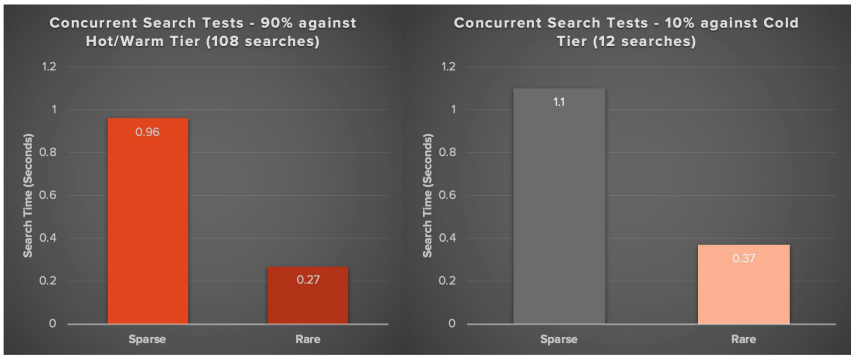

Figure 24: Run time of sparse and rare searches with 100% Hot/Warm tier with 90% against Hot/Warm tier and 10% against the Cold tier.

As you can see from the above graphs, there was no significant difference in the search time for both sparse and rare searches against 100% and 90% of Hot/Warm tier as both performed read I/Os against FlashArray. In the case of the searches that went after the 10% of Cold tier on FlashBlade, the response time was little higher than that of the Hot/Warm tier but very much comparable considering each sparse search was going after 180GB of the dataset and still completes the search in 1.1 seconds as opposed to 0.96 seconds per search when it was against the Hot/Warm tier.

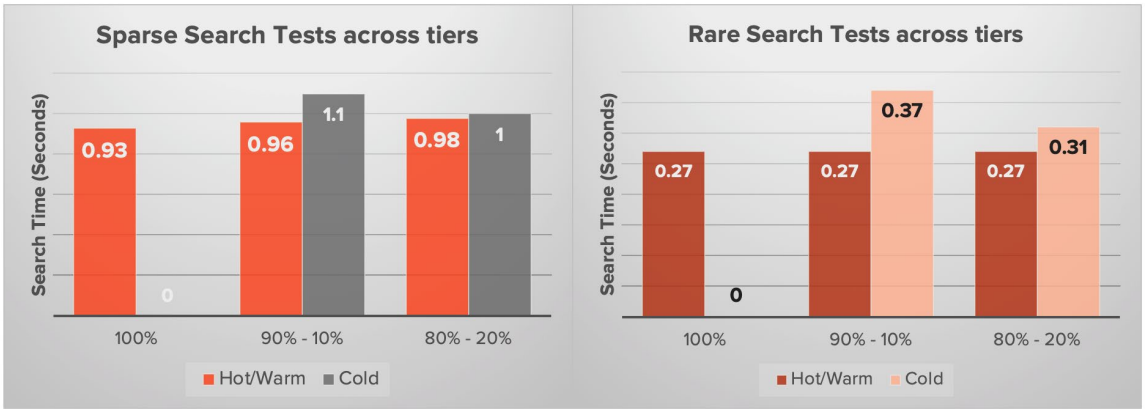

Figure 25: Rare vs. sparse search tests across tiers.

It is evident from the search test results illustrated in the graphs above that the search performance against cold tier is almost similar to that of the Hot/Warm tier irrespective of the type of searches. This revalidates the notion that a cold tier is not cold anymore when they are placed in Pure Storage's all-flash storage systems.

The CPU utilization of the indexer cluster during all these concurrent searches did not cross over 7%. The following graph shows the CPU utilization of all six tests.

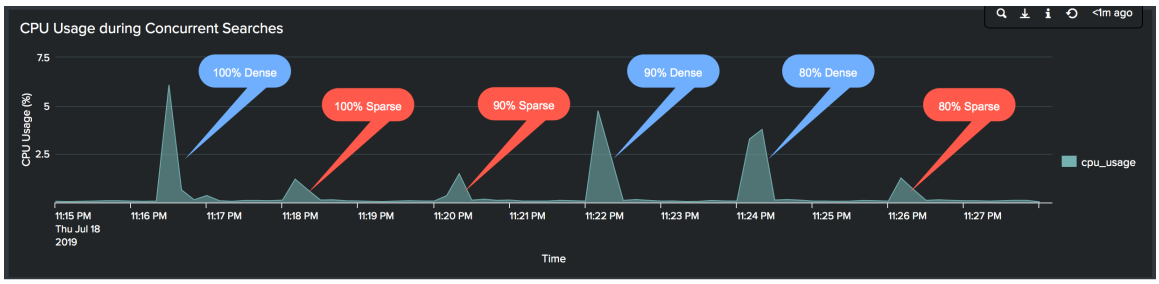

Figure 26: CPU utilization over the six tests.

Overall the Splunk Enterprise offers an accelerated search performance across all tiers within Splunk when using FlashArray for the Hot/Warm tier and FlashBlade for the Cold tier.

Best Practices

Pure FlashArray

FlashArray Volumes

- Always create a separate FlashArray volume for every indexer to host the Hot/Warm tier.
- As Pure FlashArray volumes are always thin-provisioned, Splunk Administrators can provision a large-sized volume to avoid adding additional volumes to meet the space growth.
- Keep all the FlashArray volumes for all the indexers in a cluster at the same size.

Logical Volume Manager

It is recommended you use the logical volume manager (LVM) at the indexer level to attach the FlashArray volume to a volume group and carve out the logical volume for the Hot/Warm tier out of it. This enables dynamic storage addition when the indexer needs more storage space for the Hot/Warm tier.

Linux Mount options on FlashArray volumes

We validated the use of both EXT4 and XFS filesystems for Splunk indexers. As buckets age and when directories are removed, use the DISCARD mount option to issue the TRIM command to Pure FlashArray to release the space occupied by those directories. The following are the recommended mount options:

```
discard,noatime
```

If the discard option is not a preferred option based on your standard operating procedure, make sure to issue the **fstrim** command periodically, like once a day or once a week to release the space at the FlashArray level.

Linux

The Linux recommended settings for Pure FlashArray is documented under the Solutions page at Pure Storage's support site. Please follow the same when attaching FlashArray volumes to the indexer nodes that use Linux.

Pure FlashBlade

FlashBlade Filesystems

- Always create a separate NFS filesystem for every indexer to host the Cold tier.
- As FlashBlade filesystems are always thin-provisioned, Splunk Administrators can provision a large filesystem to avoid updating the size to meet the space growth.
- Do not set a hard limit parameter for the filesystem size as this will limit the flexibility of adding more space as needed.
- Keep all the NFS filesystems for all the indexers in a cluster at the same size.

Linux Mount options

Use the following mount options to mount the NFS filesystem on the indexer nodes for the Cold tier:

```
rw,bg,nointr,hard,tcp,vers=3,rsize=16384
```

Do not specify the wsize option as the host can get the default size offered by FlashBlade (512K).

Splunk Configuration

Bucket Size

Splunk has predefined sizes for the bucket that can be configured under the maxDataSize parameter in indexes.conf

```
maxDataSize = <positive integer>|auto|auto_high_volume
```

The default is “auto” at 750MB whereas auto_high_volume is 10GB on 64-bit systems and 1GB on 32-bit systems.

The general recommendation by Splunk for high volume environment is to set the bucket size to auto_high_volume.

Recommended setting: maxDataSize = auto_high_volume (for Enterprise scale).

TSIDX Reduction

As Pure Storage systems offer further data reduction, capacity should not be a concern and so the recommendation is to use the TSIDX to its full benefit. Hence do not set the parameter enableTsidxReduction to “true”.

Recommended setting: enableTsidxReduction = false

Bloom Filters

Bloom filters in Splunk play a key role in the search behavior and the recommendation is to enable them.

Recommended setting: createBloomfilter = true.

Conclusion

Splunk is the market leader in the SIEM (Security Information and Event Management) and ranked #2 in the ITOM (IT Operations Management). The usage of Splunk has been growing considerably across various Organizations who also wanted to add more data sources while retaining the data for a longer period which stresses the Splunk infrastructure. Splunk Enterprise on Pure Storage systems is designed not only to provide a scalable architecture for Organizations to achieve them but also to meet the following benefits:

- Scale servers and storage independently for improved asset utilization and reduced TCO.
- Improve ingest performance with best of breed infrastructure components.
- Improved operational efficiencies like adding storage space with disaggregated storage.
- FlashArray and FlashBlade offers further data reduction through compression.
- FlashArray and FlashBlade offers additional security through encryption at rest.
- Splunk Enterprise on Pure Storage is architected for massive scale.

Splunk Documentation

- Capacity Planning Manual: Summary of performance recommendations
- Capacity Planning Manual: Reference hardware
- Capacity Planning Manual: How Splunk Enterprise calculates disk storage

Appendicies

Appendix A: Splunk Enterprise Components

An **Index** is a collection of databases that are subdirectories. Splunk manages all its index data in the flat file format and doesn't use any sophisticated database management systems.

An **Indexer Cluster** is a group of indexers configured to replicate each other's data so that the system keeps multiple copies of all data to prevent data loss while enabling data availability for searching in case of an indexer node failure. As the indexer cluster features automatic failover, in case of an indexer failure Splunk automatically fails that over indexer to the next indexer, enabling the incoming data to be indexed and available for searching.

Search Head handles incoming search requests. In a distributed search environment, the search head sends requests to a group of indexers, a.k.a. "search peers", which perform the actual searches on their indexes and return the results. The search head merges the results and presents them to the user. Dedicated search heads have no indexes dedicated to performing search management functions such as consolidate and display.

A **Search Head Cluster** is a group of interchangeable and highly available search heads, aka Cluster Members, that shares configurations, job scheduling, and search artifacts. By increasing concurrent user capacity and eliminating single points of failure, search head clusters enable highly available and scalable search services.

A **Forwarder** is a small-footprint version of a Splunk instance that forwards data to remote indexers for data processing and storage.

Cluster Master or **Master Node** is another instance that regulates the functioning of an indexer cluster.

Deployer is a Splunk Enterprise instance that distributes apps and other configurations to the search head cluster members. Deployer cannot be run on the same instance as a cluster member.

Monitoring Console is the Splunk Enterprise monitoring tool that allows you to view detailed performance information about your Splunk Enterprise deployment through predefined dashboards. Some of the available dashboards are indexing performance, index and volume usage, license usage, search performance, search head, and indexer clustering, etc.

License Master controls one or more license slaves and is generally used when you have more than one indexer and want to manage indexer access to purchased license capacity from a central location.

Deployment Server is a Splunk Enterprise instance that acts as a centralized configuration manager. It is the tool for distributing configurations, apps, and content updates to groups of Splunk Enterprise instances. This is generally used to update forwarders, non-clustered indexers, and search heads. This is not a required component to manage forwarders and other instances. Based on your preference and/or standard operating procedures, tools like Chef, Puppet, Salt, etc., can be used.

Appendix B: Installing Cluster Shell Utility

The Cluster Shell (**clush**) utility enables system and Splunk administrators to be more productive in managing Linux clusters by executing commands in parallel on a cluster. It can execute commands interactively or within shell scripts. A **clush** utility requires ssh.

To install **clush**:

1. Install the EPEL (Extra Packages for Enterprise Linux) repository on the central server used for managing other servers.
2. Install cluster shell utility.

   Note: In our setup, EPEL was set up on the **splunk-cm01** server and used for managing all other Splunk nodes.

```
[root@splunk-cm01 ~]# wget https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
[root@splunk-cm01 ~]# yum install -y epel-release-latest-7.noarch.rpm
```

3. Set up an SSH access (without a password) to all nodes being managed by the central server through clush.
4. (Optional) If you want to manage all nodes as splunk user, do the following as splunk user. In our instance, we used the root user.
   a. Configure an SSH public key for access without a password by entering ssh-keygen command.
   b. Run ssh-copy-id command to copy the id_rsa.pub file across all the nodes.

```
[root@splunk-cm01 ~]$
Generating public/private rsa key pair.
Enter file in which to save the key (/home/splunk/.ssh/id_rsa): Enter passphrase (empty
for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/splunk/.ssh/id_rsa. Your public key has been
saved in /home/splunk/.ssh/id_rsa.pub. The key fingerprint is:
66:5b:97:cc:5b:3e:f2:8a:d4:4a:91:11:88:e6:89:34 splunk@splunk-admin1 The key's randomart
image is:
+--[ RSA 2048]----+
|       . ..    |
|       E o . .|
|       . = . .|
|       . o  = .       |
|       S + = .|
|       o + +   |
|       . o + o|
|       o o .   |
|       o ...   |
+-----------------+
[root@splunk-cm01 ~]$
```

```
[root@splunk-cm01 ~]$ for host in cm01 ix01 ix02 ix03 ix04 ix05 ix06 ix07 ix08 sh01 sh02
sh03;
do
echo -n "$host -> ";
ssh-copy-id -I ~/.ssh/id_rsa.pub splunk-$host;
done
```

5. Configure the group details for the cluster shell by editing the following file with the node details:

```
vi /etc/clustershell/groups
all: splunk-cm01,splunk-ix[01-08],splunk-sh[01-03]
ixs: splunk-ix[01-08]
shs: splunk-sh[01-03]
```

6. Verify the cluster shell utility works by running a command. If you want to run the command only on indexer nodes, you can use the "-g indexers" argument, which limits the command to only the group specified.

```
[root@splunk-cm01 local]# clush -a hostname
splunk-ix07: splunk-ix07
splunk-ix03: splunk-ix03
splunk-ix01: splunk-ix01
splunk-ix04: splunk-ix04
splunk-ix02: splunk-ix02
splunk-ix08: splunk-ix08
splunk-ix06: splunk-ix06
splunk-ix05: splunk-ix05
splunk-sh01: splunk-sh01
splunk-cm01: splunk-cm01
splunk-sh03: splunk-sh03
splunk-sh02: splunk-sh02
[root@splunk-cm01 local]# clush -g shs hostname
splunk-sh01: splunk-sh01
splunk-sh03: splunk-sh03
splunk-sh02: splunk-sh02
```

For more information about ClusterShell, go to http://clustershell.readthedocs.io/en/latest/.

Appendix C: Useful Splunk Searches

Hot Tier Space Usage

```
| dbinspect index=<index-name> cached=false | search state=hot | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024 |fields diskinGB
```

Warm Tier Space Usage

```
| dbinspect index=<index-name> cached=false | search state=warm | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024
```

Hot/Warm Tier Space Usage

```
| dbinspect index=* cached=false | stats sum(sizeOnDiskMB) AS diskTotalinMB by state
```

Bucket Details of an Index

```
|dbinspect index=<index-name>|eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S") |fields start_time, end_time, state, bucketId, eventCount
```

Summarized Bucket Details by Index That Are Participating in the Search by the Date Range

The following search is looking for bucket details that have data within the date range of 05/20/19 08:00:00 and 05/20/19 20:00:00. Please modify the index names and date range to meet your needs.

```
|dbinspect index=apache-pure* |dedup bucketId| eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S")
| where ("05/20/19 08:00:00" >= start_time AND "05/20/19 08:00:00" <= end_time) OR ("05/20/19
```

Appendix C: Useful Splunk Searches

Hot Tier Space Usage

```
| dbinspect index=<index-name> cached=false | search state=hot | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024 |fields diskinGB
```

Warm Tier Space Usage

```
| dbinspect index=<index-name> cached=false | search state=warm | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024
```

Hot/Warm Tier Space Usage

```
| dbinspect index=* cached=false | stats sum(sizeOnDiskMB) AS diskTotalinMB by state
```

Bucket Details of an Index

```
|dbinspect index=<index-name>|eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S") |fields start_time, end_time, state, bucketId, eventCount
```

Summarized Bucket Details by Index That Are Participating in the Search by the Date Range

The following search is looking for bucket details that have data within the date range of 05/20/19 08:00:00 and 05/20/19 20:00:00. Please modify the index names and date range to meet your needs.

```
|dbinspect index=apache-pure* |dedup bucketId| eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S")
| where ("05/20/19 08:00:00" >= start_time AND "05/20/19 08:00:00" <= end_time) OR ("05/20/19 20:00:00" >=
start_time AND "05/20/19 20:00:00" <= end_time) OR ( start_time >= "05/20/19 08:00:00" AND end_time <=
"05/20/19 20:00:00")|stats sum(eventCount) as ec sum(sizeOnDiskMB) as mb count as Buckets by index |eval
diskinGB=round(mb/1024,2), EventCount=tostring(ec, "commas") |fields index, EventCount, diskinGB, Buckets.
   )
```

©2021 Pure Storage, the Pure P Logo, and the marks on the Pure Trademark List at https://www.purestorage.com/legal/productenduserinfo.html are trademarks of Pure Storage, Inc. Other names are trademarks of their respective owners. Use of Pure Storage Products and Programs are covered by End User Agreements, IP, and other terms, available at: https://www.purestorage.com/legal/productenduserinfo.html and https://www.purestorage.com/patents.

The Pure Storage products and programs described in this documentation are distributed under a license agreement restricting the use, copying, distribution, and decompilation/reverse engineering of the products. No part of this documentation may be reproduced in any form by any means without prior written authorization from Pure Storage, Inc. and its licensors, if any. Pure Storage may make improvements and/or changes in the Pure Storage products and/or the programs described in this documentation at any time without notice.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. PURE STORAGE SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

Pure Storage, Inc.
650 Castro Street, #400
Mountain View, CA 94041