PURESTORAGE®
Uncomplicate Data Storage, Forever

WHITE PAPER

Accelerating Splunk at Enterprise Scale with FlashArray

A framework for accelerating Splunk with Pure Storage FlashArray.

Contents

Introduction

Splunk Enterprise has long been a pioneer in the Security Information and Event Management aka SIEM and a leader in the data platform industry helping companies solve their complex data management and network security challenges.

Even such a mission critical application like Splunk is not immune to the challenges caused by the exponential data growth. Splunk Enterprise's long-established indexer cluster deployment with direct attached storage provides high data availability and fidelity but presents significant operational challenges as data volumes grow.

Some of the common challenges encountered by the customers are:

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

Pure Storage®, a pioneer of the all-flash array industry, has helped numerous organizations reduce the complexity of their storage management, making their infrastructure team more agile and efficient. Splunk Enterprise on Pure Storage FlashArray™ accelerates access to critical, real-time data while increasing availability, reducing overhead costs, and improving operational efficiencies.

Goals

This white paper explains the benefit of deploying Splunk Enterprise on Pure Storage FlashArray and showcases the resulting performance and operational improvements. The white paper also covers the best practices for Splunk on FlashArray.

Audience

This white paper is intended 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 Pure Storage FlashArray. A working knowledge of Splunk, Linux, server, storage, and networks is assumed but is not a prerequisite for understanding this document.

Challenges for Enterprises

Challenge #1: Complexity Grows with Scale on Direct 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 indexer'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 and reliability.

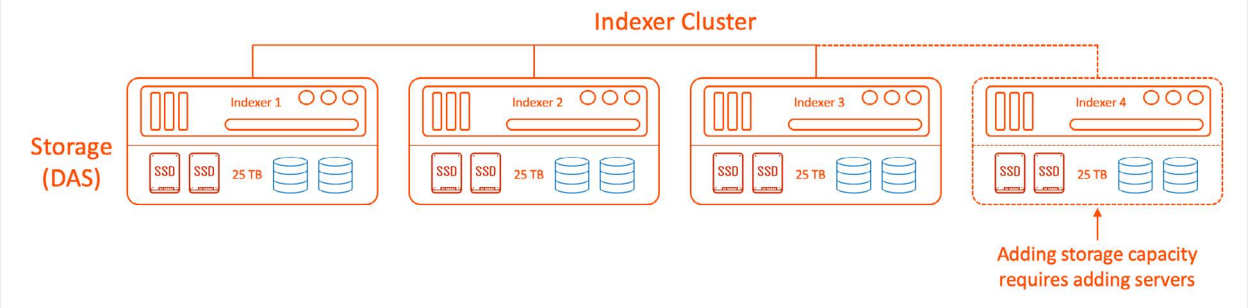

FIGURE 1 Complexity with direct attached storage

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.

Solution #1: Enterprise Storage to address the complexity with DAS

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

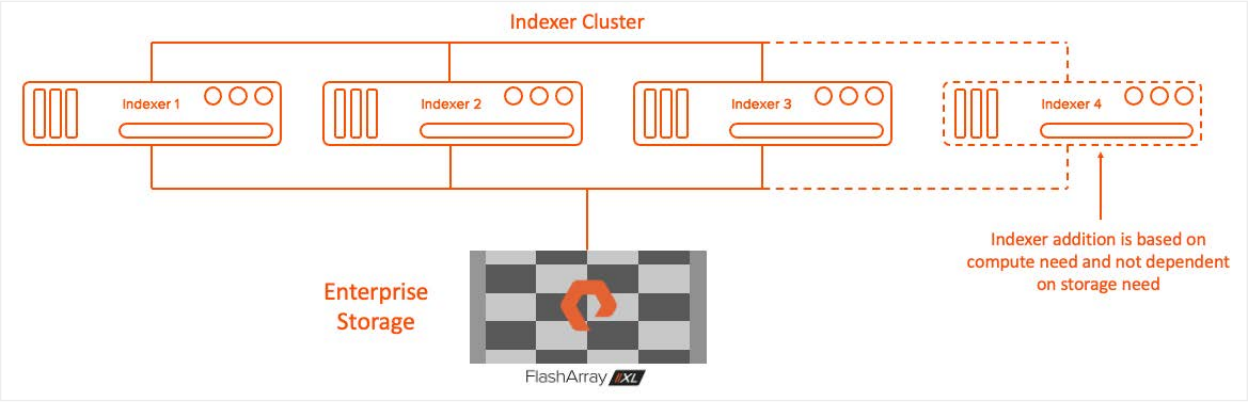

FIGURE 2 Benefits of disaggregating storage from compute

This means storage space can be added independently by adding storage shelves into the FlashArray as needed rather than adding more indexer nodes as in the traditional DAS model. This results in a reduction of indexer nodes which can help reduce the total cost of ownership and complexity associated with managing numerous indexers.

Challenge #2: Increased storage usage & inconsistent search performance

A key requirement of a distributed scale-out model is the copy or replica of the data on the additional indexer nodes based on the replication factor. An indexer cluster with a replication factor (RF) of 3 and a search factor (SF) of 2 requires the raw data to be replicated to two additional indexer nodes while the index data to be replicated to an additional indexer node on top of the indexer node where it got ingested. In addition to this, the high availability requirements on the DAS storage warrants hardware or software level RAID which further increases the total storage usage. Even though Splunk compresses the raw data, still having multiple RF and SF copies across the indexer nodes spikes up the space usage considerably in an indexer cluster environment.

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 came at the cost of slow searches.

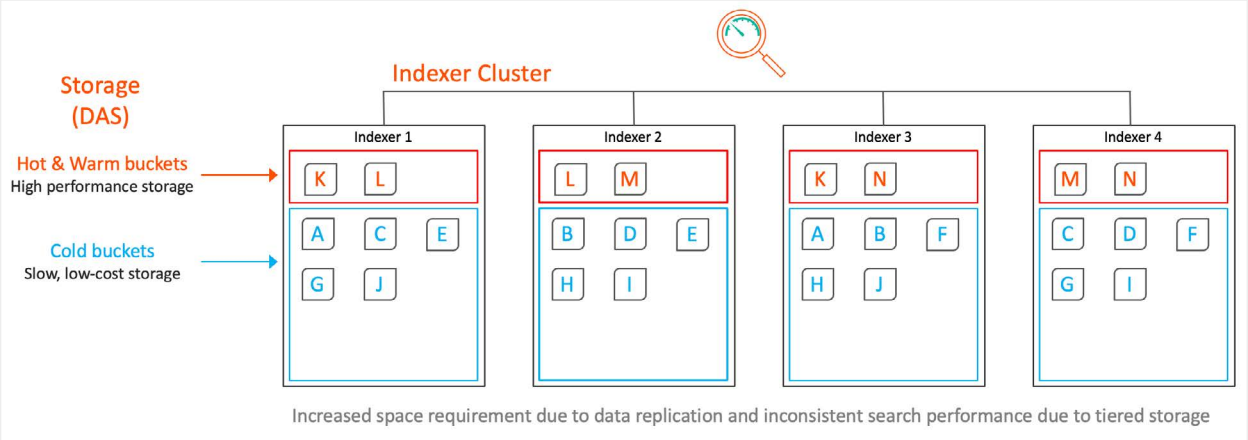

FIGURE 3 Increased storage usage due to data replication

With 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. IBM says it takes an average of 280[1] days to detect a data breach which means older data must be searched in case of a security incident and the searches should be faster. Hence having historical data or cold data on traditional cheap-and-deep disk storage is inadequate as the search performance is poor or inconsistent.

Solution #2: Space efficient All-Flash Storage with real time search performance

Pure Storage FlashArray is a thin-provisioned storage system that offers inline data services like deduplication and compression that reduces the storage requirements with the indexer cluster deployment of Splunk Enterprise. As the Splunk compressed raw data files between nodes, to meet the replication factor, have the same exact copy, they will benefit out of the FlashArray's deduplication. Meanwhile the time series index files (tsidx) are ASCII type in nature, they benefit from the compression at the storage level. Overall, FlashArray's data services reduce the storage requirements of Splunk.

The added benefit of enterprise storage like FlashArray is the in-built data protection through software RAID that reduces the operational overhead of the system administrators in manually creating the RAID every time an indexer is added for additional storage. On top of the data protection, FlashArray offers an always-on encryption service that keeps the data encrypted at rest—addressing the compliance requirements for customers.

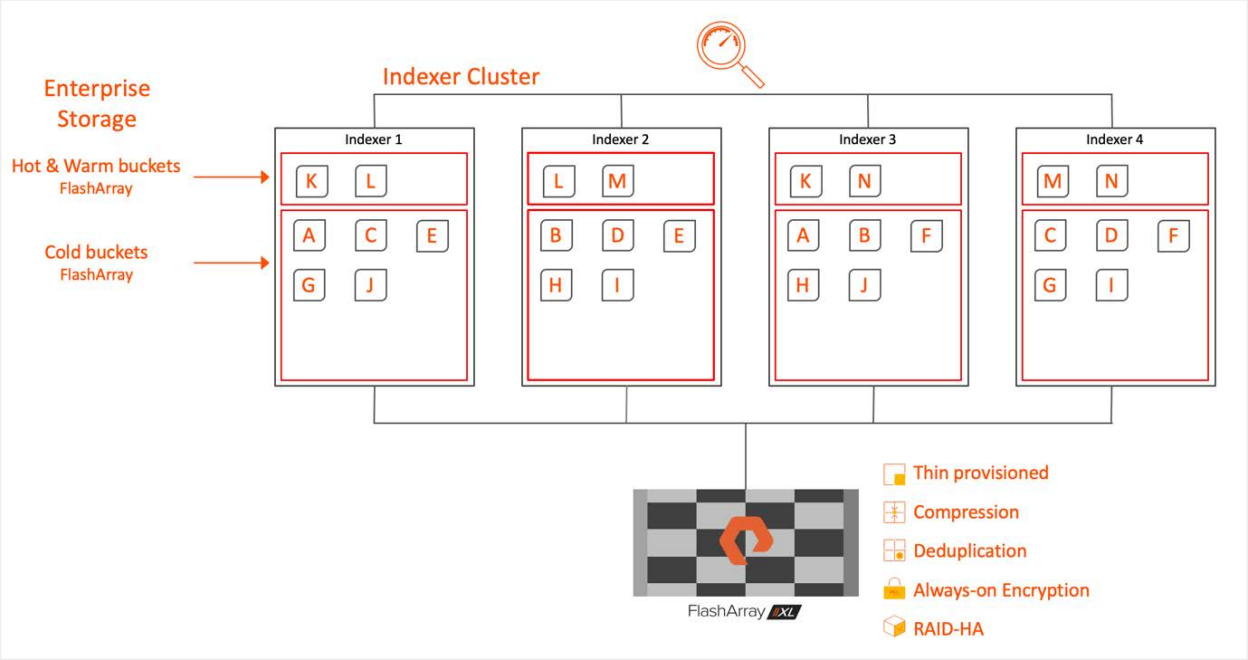

FIGURE 4 Splunk Enterprise on FlashArray with hot/warm and cold tiers

The all-flash technology of FlashArray delivers better performance for the Splunk functions like data ingest and searches in comparison to that on DAS. The performance characteristics of the cold tier now match that of hot/warm tier and the only distinction between hot/warm and cold bucket is how Splunk ages the data on those buckets. Customers can search any available data across hot/warm and cold tiers and get similar and consistent performance, thus eliminating any performance difference due to tiered storage.

Executive Summary of Benefits

This white paper highlights the benefits of running Splunk Enterprise on FlashArray. FlashArray is used as the backend block storage for the Splunk hot/warm and cold tiers. This white paper demonstrates how running Splunk on FlashArray 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.

- Splunk users can feel exasperated while searching for critical data on a traditional cold tier hosted on direct-attached hard drives.
- The Splunk Enterprise on FlashArray provides Splunk users with a consistent search experience across tiers with an all-flash storage system for hot/warm, and cold 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 operational complexity at scale. More indexers mean more patching, rebalancing, upgrading and possibility of user errors.

- With FlashArray replacing the DAS for hot/warm and cold tiers, customers can now consolidate storage across indexers. Disaggregating storage from compute enables independent scaling of compute and storage and thus reducing TCO significantly.
- Customers can consolidate storage across their entire data pipeline, including analytics apps such as Kafka, Spark, Vertica etc. Thin provisioned storage allows the customers to reduce and consolidate storage requirements by sharing storage across servers and platforms.

Enjoy the added data reduction through FlashArray's inline deduplication and compression on top of the Splunk compression. Splunk Enterprise on FlashArray with RF/SF=2 can benefit data reduction starting at 2:1.

NOTE: *Results vary with the cardinality of data. Higher RF/SF can yield better data reduction on FlashArray.*

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 FlashArray, administrators can:

- Reduce dramatically the time taken to rebalance data during indexer adds for additional storage. In our testing, Splunk on FlashArray reduced the time taken to add storage from 11 hours and 45 minutes to under 60 seconds and avoided impact to search.
- Get built-in data protection and reduced administrative overhead with built-in RAID, and always-on encryption.
- Forget about sizing the storage correctly and pay only for what is used with Pure-as-a-Service™. Help save the planet with Splunk, by reducing power consumption compared to DAS architectures.
- Never worry about upgrading their storage with Pure Storage Evergreen® Storage.

Technology Overview

Unified Block and File on FlashArray

Pure Storage FlashArray is a software-defined unified block and file storage product. It offers an effortless and consistent experience and behaves in an efficient manner by offering data reduction without an impact on performance. All Pure Storage products offer an Evergreen® product model to increase capacity and performance without the need to purchase new storage products. Lastly, FlashArray enables businesses and organizations to drive down direct carbon usage in their data storage systems by up to 80% compared to competitive all-flash systems and even more against magnetic disk.

The FlashArray product line caters towards multiple business needs and use cases with these distinct offerings:

- FlashArray//E™: 80% less energy at 60% less cost for active data archives, file repositories and other use cases
- FlashArray//C™: An all-QLC FlashArray with consistent performance at 2-4ms latency for capacity-oriented workloads
- FlashArray//X™: Provides latency as low as 150µs to power critical applications and business operations
- FlashArray//XL™: Enterprise-grade performance and scalability for demanding workloads

FIGURE 5 Pure Storage FlashArray models

Built on the Proven FlashArray Platform with Purity Data Services

Move beyond basic SSD with DirectFlash®: FlashArray™ moves beyond legacy SSD and architectures made to have flash pretend to be hard disk. DirectFlash, the world's first software-defined flash module, enables the Purity software to speak directly to raw NAND with a super-efficient NVMe protocol for even faster storage network speeds between the FlashArray unit and application servers. With FlashArray//XL, Pure Storage introduces DirectFlash Module with built-in non-volatile RAM (DFMDs). DFMDs reduce rack space requirements by removing dedicated NVRAM slots from the main array chassis. This change allows NVRAM to scale with capacity and improves NVRAM throughput, resulting in increased performance per rack unit and increased storage density within the array chassis.

DirectFlash Shelf: Enables addition of NVMe capacity beyond the FlashArray chassis. DirectFlash Shelf connects to the FlashArray storage via NVMe-oF protocol with RDMA over converged ethernet (RoCE), leveraging 50GB-per-second Ethernet. The shelf maintains the ability to support different sizes of DirectFlash Modules as flash density improves and new forms become available, such as SCM, QLC, and others.

DirectFlash Fabric: DirectFlash Fabric lowers network latency dramatically and enables enterprise-class reliability and data services via shared storage versus DAS. NVMe-oF enables massive optimization between the storage controllers and host over fast networking: FibreChannel, RoCE, and TCP. DirectFlash Fabric delivers greater performance and efficiency gains, including offloading the host CPU.

Cloud-based management with Pure1®: The Pure1 data management platform provides a single view to monitor, analyze, and optimize customer's storage from anywhere in the world, and it delivers alerts directly to their phone. The artificial intelligence for IT operations (AIOps) and full-stack monitoring in Pure1 help prevent, identify, and resolve high-severity outages and other critical issues. Its Workload Planner can predict array capacity and performance as well as model existing and new workloads, while Pure1 makes it simple to purchase new or additional services directly from its user interface.

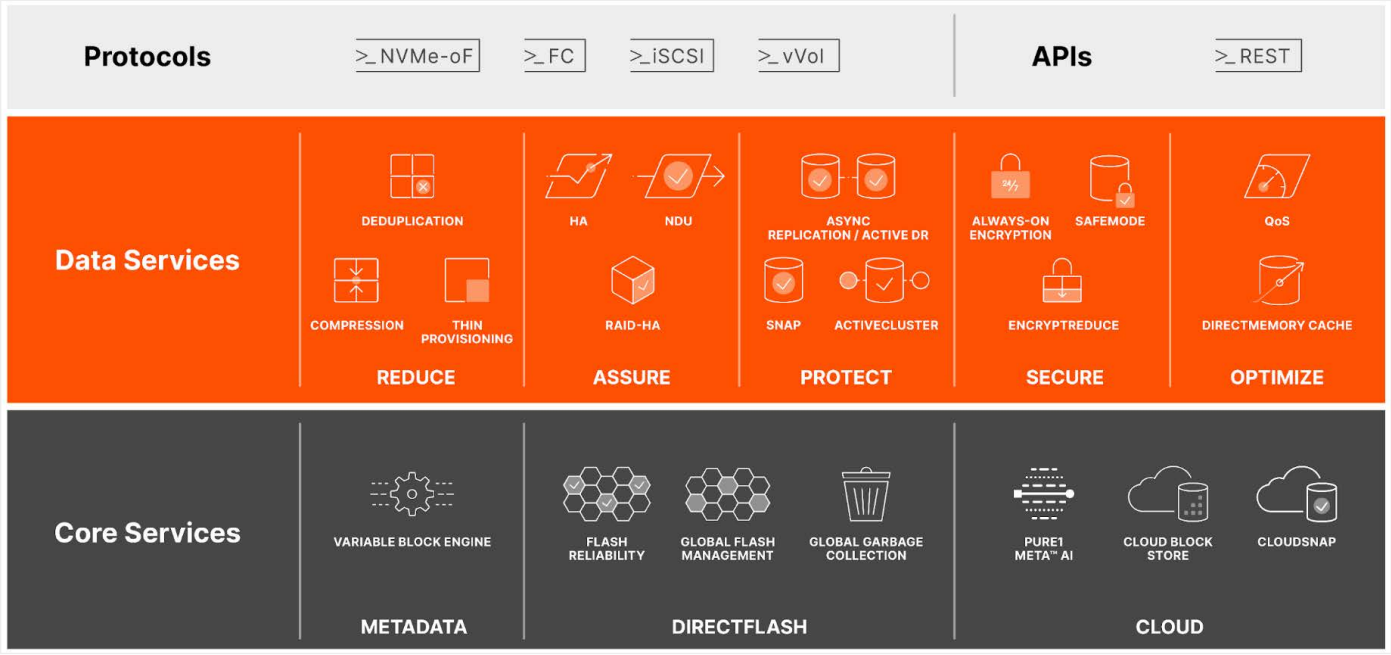

FIGURE 6 Purity FlashArray features

A Powerful Evolution of the FlashArray Family

Increased capacity and performance: The larger 5U chassis of FlashArray//XL is designed for today's higher-powered multi-core CPUs, which allows FlashArray//XL to increase performance over our FlashArray//X models. The larger 5U XL chassis provides more space for fans and airflow, which improves cooling efficiency, and for wider controllers that enable performance to scale today and well into future generations of FlashArray//XL. With greater storage density, FlashArray//XL supports up to 40 DirectFlash Modules in the main chassis. To increase capacity further, up to two DirectFlash Shelf expansion shelves can be connected.

Increased connectivity, greater reliability, and improved redundancy: FlashArray//XL doubles the host I/O ports compared to FlashArray//X, for up to 36 ports per controller, and the //XL model provides more expansion slots for configuration flexibility. It doubles the bandwidth for each slot, including full bandwidth for mixed protocols. FlashArray//XL offers multiple 100GbE RDMA over Converged Ethernet (RoCE) links that are very robust to hot-plug and provide faster controller failover speed. The RoCE controller links also offer Increased resiliency capabilities, including improved "cross-controller" high availability (HA) with a nearly three-times increase in bandwidth, more stability at high array load, minimal disruption during failover, and four power supplies that operate in an N+2 configuration.

DirectFlash Modules with distributed NVRAM: DirectFlash Modules include onboard distributed non-volatile random-access memory (DFMD). Separate NVRAM modules are no longer required. With DFMD, NVRAM capacity, NVRAM write bandwidth, and array capacity scale with the number of DFMDs, lifting the limit on write throughput.

DirectCompress Accelerator: Included with every FlashArray//XL shipment, the DirectCompress Accelerator (DCA) increases compression efficiency by offloading inline compression to a dedicated PCIe card. It ensures maximum compression rates, even when the system is under a heavy load, and stretches capacity to reduce overall storage costs and to extend the value of FlashArray//XL.

Splunk Enterprise

Splunk Enterprise makes it simple to collect, analyze and act upon the value of the big data generated by the technology infrastructure, security systems, and business applications, giving customers 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 customer's 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.

RHEL Linux

Red Hat Enterprise Linux (RHEL) is the world's leading enterprise Linux platform, certified on hundreds of clouds and with thousands of hardware and software vendors. RHEL can be optimized to run on servers or high-performance workstations, and supports a range of hardware architectures like x86, ARM, IBM Power, IBM Z, and IBM LinuxONE. RHEL includes built-in security features like live kernel patching, security profiles, security standards certification, and a trusted software supply chain to help meet today's high security and compliance expectations.

Solution Design

The solution includes twelve virtual machines and eight physical servers with local NVMe drives for hosting the Splunk Enterprise configuration. The configuration consists of a cluster Manager, three search heads and eight universal forwarders on virtual machines and eight indexers on the physical servers. The Splunk's hot, warm, and cold data are hosted on Pure Storage FlashArray//XL130 connected over NVMe-TCP.

FlashArray Configuration

FlashArray//XL hosts the hot, warm, and cold data of the Splunk indexes. Separate volumes for hot/warm and cold tiers for every indexer node were carved out of the FlashArray and attached to the respective indexer nodes as a block device over NVMe-TCP. The hot, warm, and cold data on FlashArray is deduped, compressed, and encrypted inline.

Component	Description
FlashArray Model	//XL130
Capacity	128.7 TB
Drives	30 × 4.28 TB DirectFlash Modules
Connectivity	4 × 100 Gb/s redundant Ethernet for NVMe-TCP 2 × 1 Gb/s redundant Ethernet (Management port)
Rack units	5U

TABLE 1 FlashArray configuration

Software Configuration

Component	Description
Operating System	RHEL 8.6
Splunk	9.1.1
Purity	Purity//FA 6.4.10

TABLE 2 Software configuration

Splunk Hardware Configuration

Component	Physical Server	Virtual Server
CPU	2 × 24 Cores (Intel Xeon 8260)	12 vCPUs
Memory	256 GB	128 GB
Network	2 × 100 GbE	2 × 10 GbE
Local Storage	8 × 1.6TB NVMe drives	None

TABLE 3 Splunk Hardware configuration

Logical Topology

For enterprise-grade deployment of Splunk Enterprise that requires multiple terabytes of data to be ingested daily while supporting numerous concurrent searches, a distributed deployment of indexer clustering is recommended. This solution had the following configuration:

- 1 cluster manager
- 8 Indexers in an indexer cluster setup
- 3 search heads in a search head clustering
- 8 universal forwarders

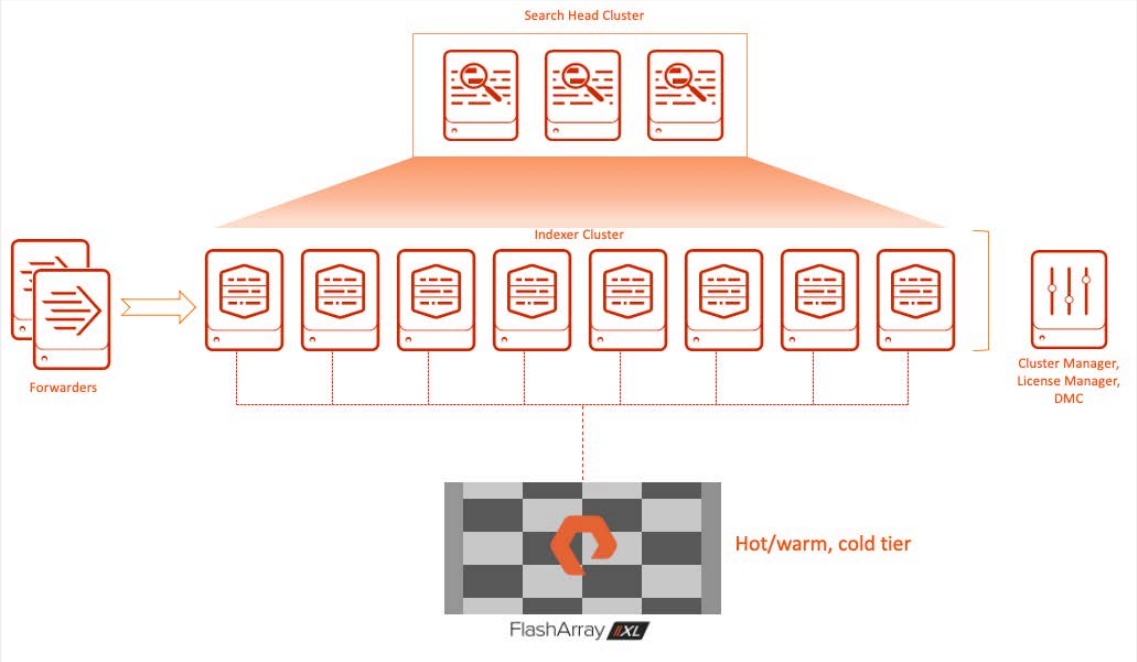

FIGURE 7 Logical Topology

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 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.

Please go over the Splunk Validated Architectures (SVAs) for stable, efficient, and repeatable splunk deployments.

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 store the replicated data along with additional processing load on indexers and network traffic to replicate the data between indexers.

There are two key factors named *replication factor* and *search factor* that determines 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 which consists of the time series 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 default value is two (2).

Storage Sizing Guidelines

To size the storage requirements for the Splunk in an indexer cluster deployment model, 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).
- Replication factor (RF).
- Search factor (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 increase in the storage sizing requirement.*

In the case of FlashArray, the storage requirement for Splunk is calculated as follows:

```
FA Storage required = daily ingest data *total retention period* 0.15+0.35*SF
```

NOTE: *These calculations are estimates and do not include the data reduction ratios offered by FA as well as the RAID overhead.*

As seen from the above calculations, the replication factor doesn't make a difference with the Hot/Warm and cold tiers calculation on FlashArray because the multiple copies of the rawdata is always deduped at the storage level, rendering extra storage space. If the deduped single copy of the raw data is a concern for being the single point of failure, be assured that FlashArray uses RAID-HA data protection that protects against concurrent dual-drive failures and initiates re-builds automatically within minutes and detects and heals bit-errors.

For example, for a daily ingest rate of 10TB per day with a total retention of 360 days (90 days in hot/warm and 270 days on cold) with RF=3 and SF=2, the total storage requirement is:

```
Storage required = 10 TB * 365 * (0.15 * 3 + 0.35 * 2) = 4,140 TB
```

Whereas in the case of FlashArray, the total storage required is:

```
Storage required with FA = 10 TB * 365 * (0.15 + 0.35 * 2) = 3,060 TB
```

The above number includes the storage savings through deduplication. The usable FlashArray storage requirement to support Hot/Warm/Cold tiers would further go down in the range of 1,800 TB and 2,400 TB based on the compressibility of the time-series index data.

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[1], 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 FlashArray is validated by testing the above key functions of Splunk Enterprise namely data ingestion, search, and the operational efficiency of Splunk space management.

1. Data Ingestion
2. Search Behavior
3. Operational Efficiency

Data Ingestion

Ingest Test Overview

The ingest test involved ingesting 50TB of data into the indexer cluster with 8 peer nodes with all indexes configured with replication factor and search factor set to 2. During the ingest, concurrent searches were performed to mimic the real world. The time taken to load the 50TB of data was captured along with the system utilization.

Ingest Test Setup

The dataset used for the ingest testing was the apache log data of sourcetype *access_combined* with additional keyword on every event that can be used for dense and rare searches. The dataset was generated through a custom script developed using Go programming language. The data generation script was run on the 8 forwarders which generated 256G of log files per day per indexer amounting to 2TB across 8 indexers per day. To ingest 50TB of data, 25 days' worth of logfiles were ingested from the eight forwarders.

To speed up the data through the forwarders, the throughput entry maxKBps in $SPLUNK_HOME/etc/system/local/limits.conf was set to 0 which eliminates any regulation on the data throughput from the forwarder to the indexers. The default throughput of universal forwarders is 256KBps. For testing purposes, a larger throughput (e.g., 10240 or 0) might be acceptable but for standard operating procedures, the customers should set a value that reflects their environment, so it doesn't overwhelm their network infrastructure. The source log files were ingested into the index **nvme-apache** from the eight forwarders using the **oneshot** method.

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

Ingest Test Results

The data ingestion of 50TB into the eight indexers took 42 hours 37 minutes 50 seconds at an average of 341.62 MBps or 28.15 TB/day.

Ingested Data	Elapsed Time	Index Rate	Indexing Rate per indexer	Daily Ingest Rate
50 TB	42h 37m	341.62 MBps	42.70 MBps	28.15 TB/day

TABLE 4 Ingest test results

During the ingest, the overall CPU utilization of all the eight indexers averaged around 10% with maximum utilization around 30% and the indexer cluster reached the peak indexing rate around 400 MBps or 50MBps per indexer.

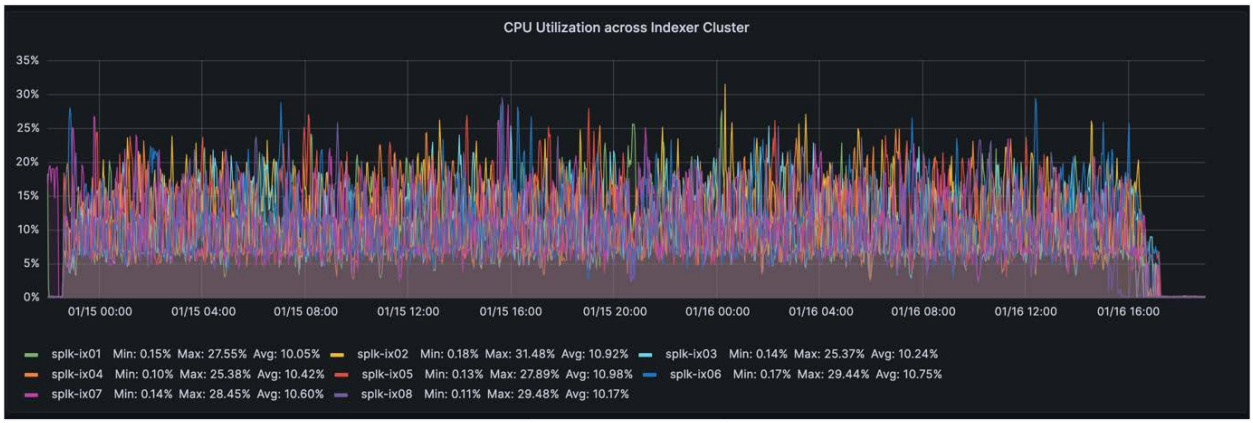

FIGURE 8 CPU utilization across the indexer cluster

As the data lands on the hot buckets on Splunk a steady stream of writes went onto the hot tier hosted on the FlashArray. As the buckets were constantly rolled from hot to warm to cold buckets during the large scale of ingest, there was a steady stream of read activity from the hot/warm tier in addition to the writes across all tiers.

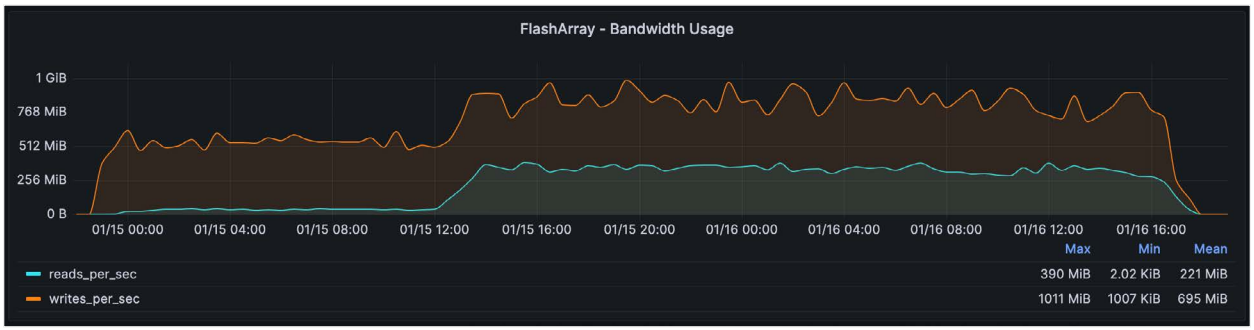

FIGURE 9 FlashArray bandwidth usage during data ingest

Based on the test results, the FlashArray was never stressed during the large scale ingest and FlashArray has a lot more processing power left to take up additional workloads.

Space Usage

Splunk compresses the incoming rawdata during ingestion and generates time series index data which takes up the space around 50% of the ingested data with a single copy (RF=1, SF=1). As the number of replicated copies goes up, so does the space usage by a factor of the copies. For example, RF=2, SF=2 will result in Splunk using almost the same amount of space as that of the total ingested data. The data reduction can be further impacted by the cardinality of the incoming data.

After the ingest of 50TB of data, the total index size across the cluster was 44,605.44 GB or 43.56 TB which was 15% lower than the actual ingested data of 50TB even after creating two copies of raw data and index data. This data when stored on FlashArray resulted in an overall space usage of 24.21 TB across hot and cold volumes resulting in a data reduction of 1.8:1 from the Splunk's space usage of 43.56 TB.

FIGURE 10 Space usage on Splunk and FlashArray

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 **2.4 TB/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.

The test results with concurrent search load and ingest on eight indexer nodes around 30% of CPU utilization along with healthy indexing queues was **28.15 TB per day**. This is because of the server hardware for indexer nodes which are made up of 2×24 CPU cores (48 cores) in total. This aligns with Splunk's high-performance reference hardware's performance capability and clearly demonstrates the possibility of improving performance on a superior infrastructure complemented by a very efficient and highly scalable storage with FlashArray.

Search Behavior

Majority of the Splunk customers perform their searches on the near-term data which generally resides on the Hot/Warm tier. Hence the Splunk best practice is to place the Hot/Warm data on a faster storage to improve the search performance which in this solution is on FlashArray.

As more and more Enterprise level customers are interested in gaining more insights from all the data they have and not just limited to the near-term data, the requirement to search the data across the cold tier has gone up. At the same time the customers are not willing to wait for a longer time for the 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. As such, the cold tier in this solution is also hosted on FlashArray which offers the same performance characteristics as that of hot/warm for searches.

Search Test Overview

To showcase the performance of FlashArray in comparison to that of direct attached storage, same source data was ingested into two indexes, one hosted on FlashArray and another hosted on the direct attached storage. As the test environment with direct attached storage did not include hard drives for the cold tier, the search tests comparison was limited to hot/warm tier for an apple-to-apple comparison.

For the search tests, the same exact dense and rare searches were performed against indexes hosted on DAS and FlashArray and the elapsed time were compared. The dense search by nature is more CPU intensive as the search would result in returning numerous events per bucket and as such most of the time is spent in decompressing the rawdata whereas the rare search is more IO intensive where the search would result in a few events per index.

Search Test Results

The dense search resulted in processing 58,976,456 (58.9M) events out of 589,866,752 (589M) events, or one event out of every 100 events while the rare search resulted in 72 events out of the 786 million events, or one in every 10 million events.

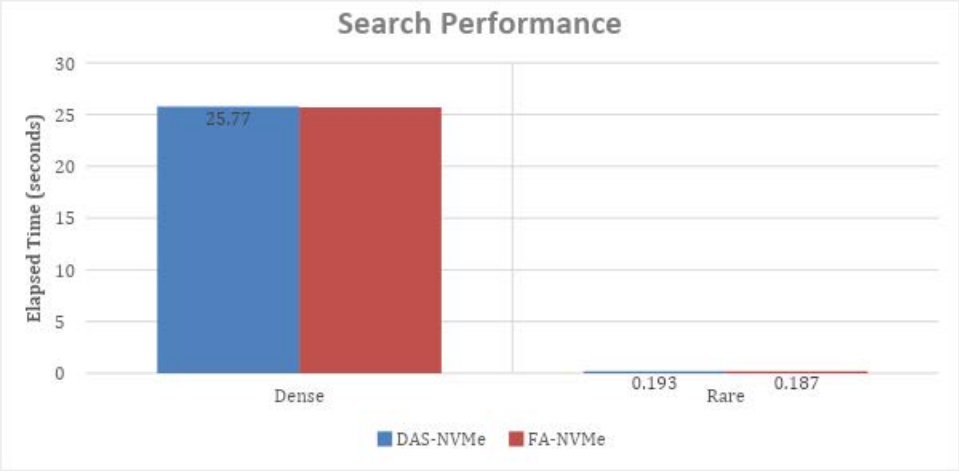

FIGURE 11 Search test results

As the chart illustrates, the search performance on FlashArray is either better or similar to that of the direct attached storage with local NVMe drives. Interestingly the searches on the cold tier on FlashArray will also exhibit the same performance characteristics as that of the hot/warm tier as the only difference is the volume names which are different logically. In fact, the same FlashArray volume for hot/warm tier can also hold the cold tier but for ease of bucket management, tracking the IO performance/usage at the hot/warm and cold tier, and for any additional data services like snapshots and replications Pure Storage recommends creating separate FlashArray volumes for hot/warm and cold tier. Regardless, the search performance against the recently ingested data in hot/warm tier or the historical data in the cold tier on FlashArray will offer consistent search performance.

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 with respect to storage management will showcase the benefits of enterprise storage like Pure Storage FlashArray in adding storage space to the indexer nodes.

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

Traditional Approach

In the traditional direct attached storage model, the indexer nodes are prepopulated with the storage devices like SSD and/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 single point of failure but at the cost of additional 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.

This 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 7-node indexer cluster had a total space of 44 TB across 3386 buckets (6772 total buckets including the replica copy). Majority of the buckets, almost 95% belong to a specific index named *nvme-apache*. In this test, an eighth indexer was added to the cluster for additional storage space and data rebalancing was performed using the following command.

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

Suggested Pure's Approach

In the disaggregated model, an 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.

In case of the Pure Storage FlashArray, it is easier and simpler to just resize the volume for each indexer peer to a bigger size, resize the multipath device and the corresponding filesystem on the respective indexers. There is no need to add a new server to get the additional space.

Operational Efficiency Test Results

Traditional Approach

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

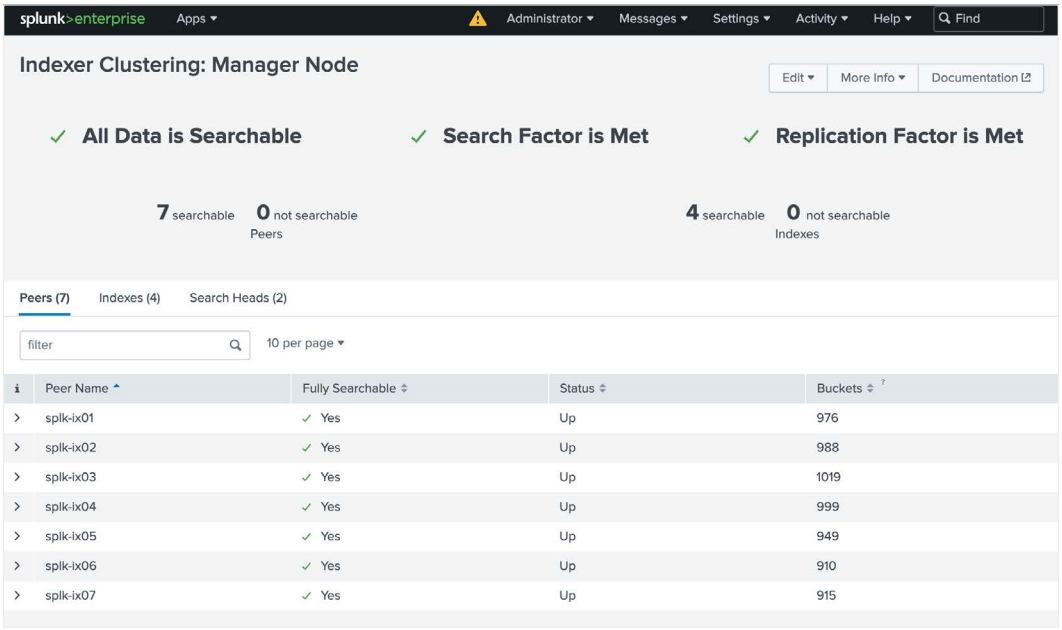

FIGURE 12 Buckets distribution prior to the data rebalance

The following dashboard shows the buckets usage on the index *nvme-apache.* The total buckets under this index were 6440 which includes the replicated buckets.

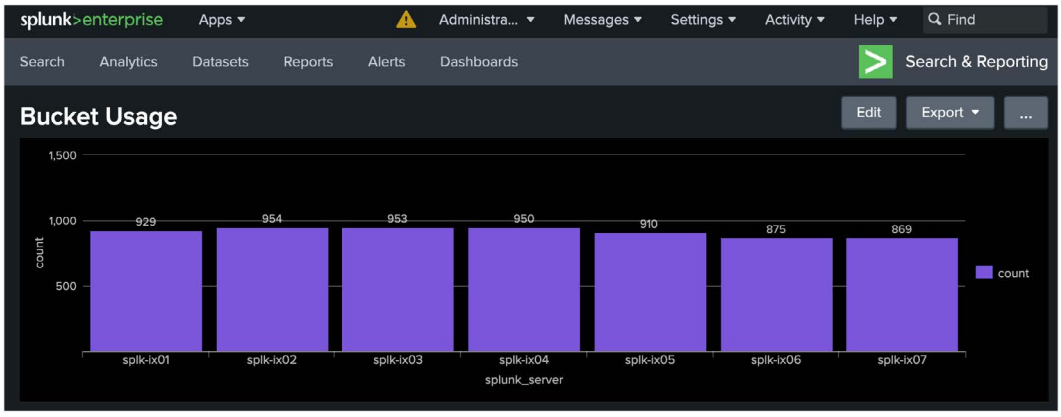

FIGURE 13 Buckets distribution of the index prior to data rebalance

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

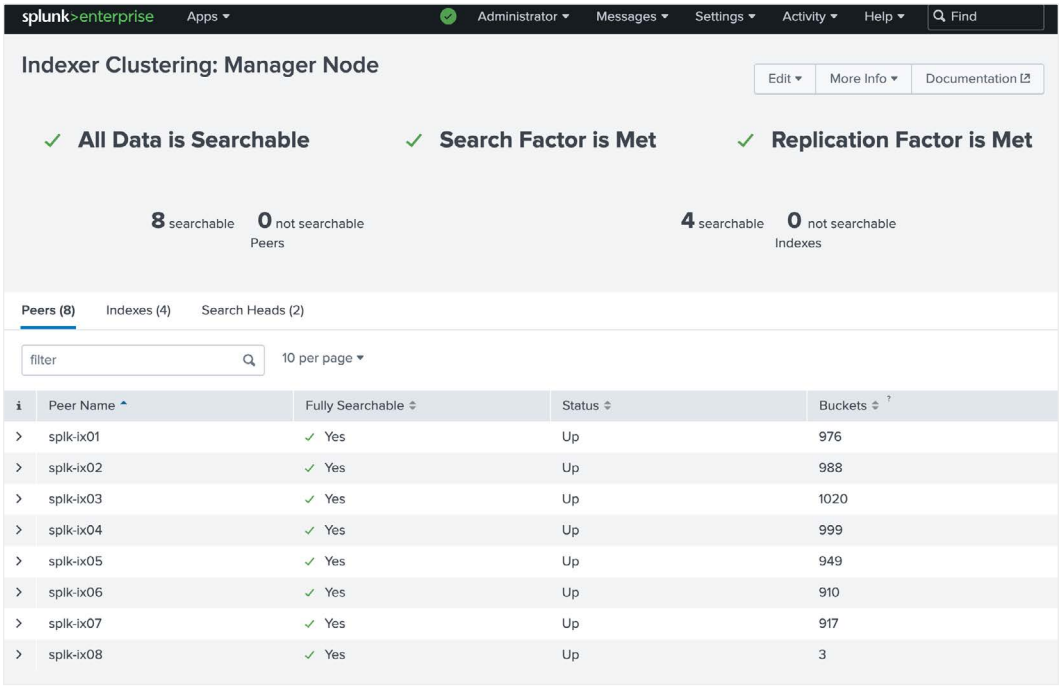

FIGURE 14 Buckets distribution after index addition

The data rebalance command was invoked through CLI which took **5 hours and 9 minutes** to complete which rebalanced around 5.5TB of data to the eighth indexer. At the end of the data rebalance the buckets were uniformly distributed across all the 8 indexers.

NOTE: *The rebalance time can change based on various factors like the searchable mode, number of buckets, network bandwidth, underlying storage medium, etc. The test environment did not include any hard drives for the Cold tier. If the indexers had hard drives, the rebalance time would have taken much longer.*

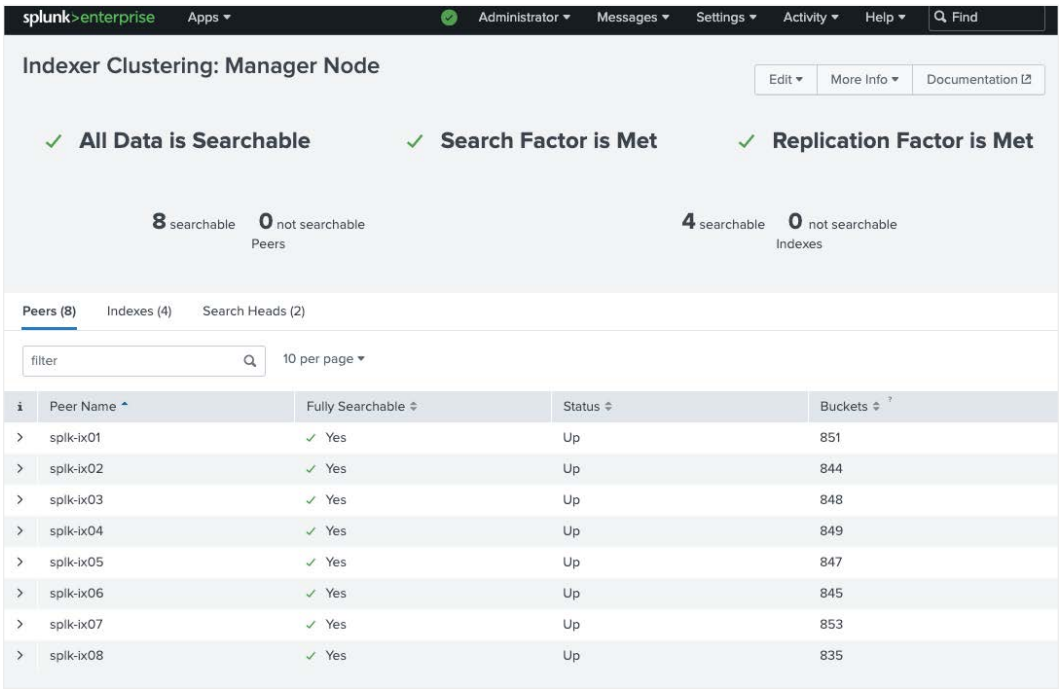

FIGURE 15 Buckets distribution after the data rebalance

The bucket fixup details during the data rebalance shows 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.

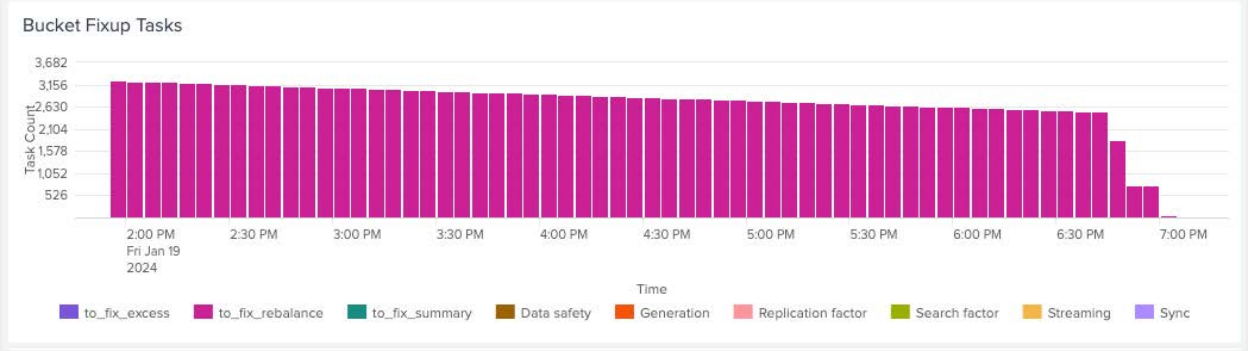

FIGURE 16 Bucket fixup activity

The following dashboard shows the bucket distribution across all indexers for the index *nvme-apache* once the data rebalance was complete.

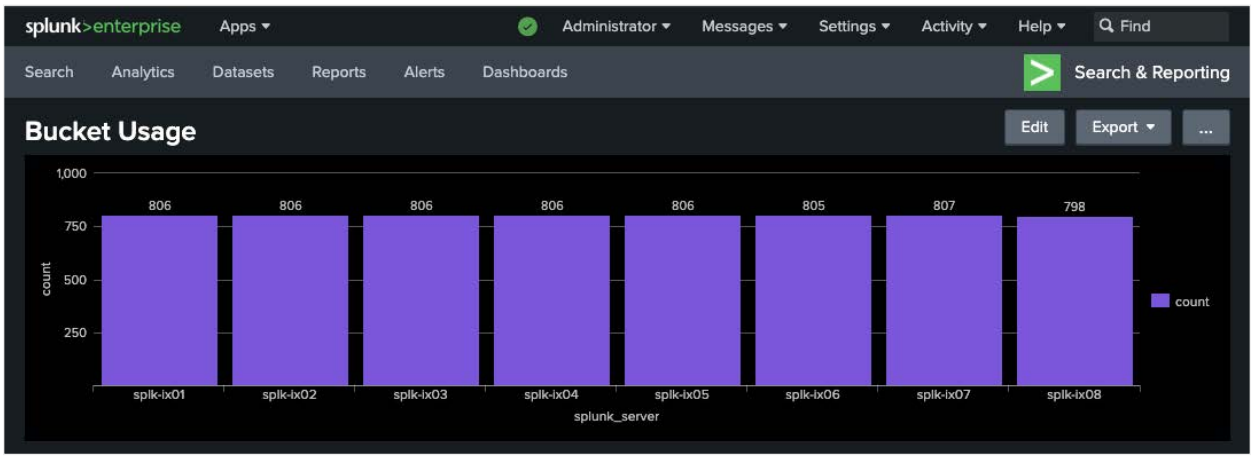

FIGURE 17 Buckets distribution of an index after data rebalance

NOTE: *Pure’s Approach to space addition*

The FlashArray volumes that is hosting the Splunk data can be resized through GUI or CLI. Following screenshot shows the nvme device details of the cold tier on the indexer prior to the resizing. While the commands to resize the cold tier were performed on all the indexers at the same time, for brevity the output is shown from one of the indexers.

```
[root@splk-ix01 ~]# nvme list |head -1 && nvme list |grep nvme8n2
Node                  SN                   Model                                    Namespace Usage                      Format           FW Rev
/dev/nvme8n2          1206D4502FE2E1B2     Pure Storage FlashArray                  131         11.00  TB /  11.00  TB    4 KiB +  0 B   G.4.10
[root@splk-ix01 ~]#
[root@splk-ix01 ~]# df -h /cold_nvme
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/splunk-cold-nvme   10T  4.1T  5.5T  43% /cold_nvme
[root@splk-ix01 ~]#
```

FIGURE 18 Space usage before FlashArray volume resize

The following FlashArray CLI command resized the splunk volume for the cold tier from 10T to 15T.

```
purevol setattr --size 15T splunk-cold-vol-ix01
```

As seen in the following figure, the *nvme list* command for the cold tier reflects the FlashArray resize right away and doesn't require any manual rescan but it doesn't reflect on the multipathed device and the filesystem until they are resized.

```
[root@splk-ix01 ~]# nvme list |head -1 && nvme list |grep nvme8n2
Node                  SN                   Model                                    Namespace Usage                      Format           FW Rev
/dev/nvme8n2          1206D4502FE2E1B2     Pure Storage FlashArray                  131         16.49  TB /  16.49  TB    4 KiB +  0 B   6.4.10
[root@splk-ix01 ~]#
[root@splk-ix01 ~]# df -h /cold_nvme
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/splunk-cold-nvme   10T  4.1T  5.5T  43% /cold_nvme
[root@splk-ix01 ~]#

[root@splk-ix01 ~]# multipath -ll splunk-cold-nvme
splunk-cold-nvme (eui.0056fae695e2004524a937ea00016c61) dm-7 NVME,Pure Storage FlashArray
size=10T features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='queue-length 0' prio=50 status=active
  |- 10:16:2:131 nvme10n2 259:13 active ready running
  |- 11:7:2:131  nvme11n2 259:15 active ready running
  |- 8:5:2:131   nvme8n2  259:9  active ready running
  `- 9:14:2:131  nvme9n2  259:11 active ready running
[root@splk-ix01 ~]#
```

FIGURE 19 Space usage after FlashArray volume resize

The following commands resize the multipathed cold tier device as well as the filesystem on Linux.

```
[root@splk-ix01 ~]# multipathd resize map splunk-cold-nvme

[root@splk-ix01 ~]# resize2fs /dev/mapper/splunk-cold-nvme
```

The below screenshot shows multipathed device resized and the filesystem resized online while Splunk was up and running and data were ingested. Both the commands took under a minute to complete.

```
[root@splk-ix01 ~]# multipathd resize map splunk-cold-nvme
ok
[root@splk-ix01 ~]# multipath -ll splunk-cold-nvme
splunk-cold-nvme (eui.0056fae695e2004524a937ea00016c61) dm-7 NVME,Pure Storage FlashArray
size=15T features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='queue-length 0' prio=50 status=active
  |- 10:16:2:131 nvme10n2 259:13 active ready running
  |- 11:7:2:131  nvme11n2 259:15 active ready running
  |- 8:5:2:131   nvme8n2  259:9  active ready running
  `- 9:14:2:131  nvme9n2  259:11 active ready running
[root@splk-ix01 ~]# resize2fs /dev/mapper/splunk-cold-nvme
resize2fs 1.45.6 (20-Mar-2020)
Filesystem at /dev/mapper/splunk-cold-nvme is mounted on /cold_nvme; on-line resizing required
old_desc_blocks = 1280, new_desc_blocks = 1920
The filesystem on /dev/mapper/splunk-cold-nvme is now 4026531840 (4k) blocks long.
[root@splk-ix01 ~]# df -h /cold_nvme
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/splunk-cold-nvme   15T  4.1T   11T  29% /cold_nvme
[root@splk-ix01 ~]#
```

FIGURE 20 Space usage after online resize2fs

The following table summarizes the storage space addition test results.

Operational Activity	Traditional Approach	Pure Storage Approach
Storage space addition	5h 9m	1 minute

TABLE 5 Operation activity test results

Pure Storage's suggested approach to resizing the Splunk volumes on FlashArray is very efficient and eliminates the need for new indexer servers just to address the additional space thus reducing the total cost of ownership.

Best practices for Splunk on FlashArray

FlashArray Volumes for Hot/Warm, Cold tiers

FlashArray has made it very simple to configure volumes for Splunk indexers. There is no need to worry about striping or RAID level, as the FlashArray's operating environment, Purity addresses them automatically.

For ease of bucket management, to enable backups of warm or cold buckets, to measure the IO performance at hot/warm, cold tiers, the recommendation is to create separate volumes for hot/warm, cold, and frozen (if the frozen data must be kept on FlashArray) tier per indexer. For example, an indexer cluster with 4 indexers will have 4 volumes for hot/warm tier and another 4 volumes for cold tier.

As the FlashArray volumes are always thin-provisioned, Splunk Administrators can provision a large-sized FlashArray volume to avoid additional FlashArray volumes to meet the space growth. Also, to avoid imbalanced space usage, keep all the FlashArray volumes for all the indexers in a cluster at the same size.

Recommended Settings:

- Always create a separate FlashArray volume for hot/warm, cold tier per indexer and separate the volume stanza for Hot/warm and cold tier in the indexes.conf file.
- Provision a large-sized FlashArray volume to avoid adding additional volumes or resizing the volumes to address the space growth.
- Keep all the FlashArray volumes for all the indexers in a cluster at the same size.

Linux Mount options

Splunk supports both EXT4 and XFS filesystems on the indexers to mount the FlashArray volumes. As the buckets age and when directories are removed, the underlying block storage must be instructed to reclaim the space using TRIM/unmap commands. This can be accomplished through the discard mount option that will issue the TRIM command to FlashArray to release the space occupied by those directories.

```
Recommended mount options: discard,noatime
```

If the discard option is not a preferred option, the **fstrim** command can be issued on the mount point periodically, once a week or a month to release the space deleted by Splunk at the FlashArray level.

Linux Best Practices

The Linux recommended settings for FlashArray are documented as a knowledge base article at the Pure Storage support site at:

https://support.purestorage.com/Solutions/Linux/Linux_Reference/Linux_Recommended_Settings

Storage Protocols

While this solution used NVMe-TCP, FlashArray//XL or any other FlashArray models that offers NVMe-oF can be configured with NVMe-FC, NVMe-TCP or NVMe-RoCE.

Meanwhile, Splunk Enterprise on FlashArray can run on other standard protocols like FC or iSCSI. NVMe is not a requirement to run Splunk on FlashArray.

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 FlashArray offers further data reduction, capacity should not be a concern and hence the recommendation is to use the TSIDX to its full benefit.

```
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

Complementing the real-time and consistent search performance across tiers with inline data reduction, encryption and operational improvements makes Splunk on FlashArray to be the ideal combination for reducing the overall total cost of ownership while scaling the business needs that includes longer retention.

Splunk Enterprise on FlashArray is designed not only to provide a scalable architecture for Organizations but also to meet the following benefits:

- Scale servers and storage independently for improved asset utilization and reduced TCO.
- Provides all flash search and consistent performance across all tiers.
- Protects Splunk data at rest through inline encryption.
- Further reduces the space usage through deduplication and compression.
- Reduces indexer servers while supporting similar or higher capacity requirements.

Additional Resources

1. Capacity Planning Manual: Reference hardware
2. Capacity Planning Manual: Summary of performance recommendations
3. Best Practices for Splunk on Pure Storage

1 https://www.ibm.com/security/data-breach

PS2563-01-en 02/24