PURESTORAGE®
Uncomplicate Data Storage, Forever

REFERENCE ARCHITECTURE

Scalable Time Series Analytics with Kx Systems kdb+ on Pure Storage FlashBlade

A reference architecture for KX Systems kdb+ on Pure Storage® FlashBlade®

Contents

Executive Summary 3

Introduction 3

- Solution Overview 3
- Solution Benefits 4

Technology Overview 5

- Compute Resources 5
- Network Resources 6
- Pure Storage FlashBlade 6
- Kx Systems kdb+ 7

Technical Solution Design 8

- Compute Layer 9
- Network Layer 9
- Data Layer 10
- kdb+ Design Considerations 10

Design Validation 11

- Configuration 12
- Results 12

Deployment 14

- FlashBlade 14
- Host Operating System (RHEL, Centos, SUSE) 17
- KX Systems Kdb+ 18

Conclusion 19

Additional Resources 19

Executive Summary

Implementing tick data analytics for quantitative trading involves several challenges, including managing the immense volume and velocity of tick data, ensuring data quality and integrity, and minimizing data latency. Substantial computational power is required to process and analyze data in real-time, necessitating scalable infrastructure and optimized algorithms. Operational challenges such as maintaining system reliability and uptime, adhering to regulatory compliance, managing costs, and ensuring security also play a significant role.

The Pure Storage® FlashBlade® portfolio was designed with data science and time-series workloads in mind. It offers differentiated value through its ability to rapidly ingest, process, and analyze extensive real-time and historical data, thereby enabling quants to make informed and timely decisions. The architecture supports seamless scalability, ensuring that as data volumes and analytical complexities increase, performance remains robust. This capability is vital for handling the high-frequency data and intense computational demands typical in trading environments.

The combination of real-time and historical data analytics empowers quants to gain immediate insights and respond swiftly to market fluctuations. Reliability and efficiency are at the core of the product, providing robust data integrity and high availability to ensure consistent and uninterrupted operations. This is particularly important in financial markets where any disruption can lead to significant financial losses.

In this reference architecture, Pure Storage FlashBlade systems provide an optimized data platform for kdb+, a leading time series database, enhancing its performance to handle high-frequency trading data with exceptional speed and accuracy. This combination accelerates various use cases, including high-frequency trading, risk management, and market research. By leveraging FlashBlade with KX Systems' kdb+, financial institutions can achieve unmatched performance, scalability, and efficiency in managing their data analytics workloads.

Introduction

This reference architecture offers a comprehensive solution for use cases such as high-frequency trading, risk management, and market research. It leverages the capabilities of Pure Storage FlashBlade to enhance the performance of the Kx Systems time series database, kdb+, accelerating tick data analytics and maximizing the historical value of data.

Design validation in this reference architecture uses a fully audited STAC-M3 benchmark, the industry standard for testing solutions that enable high-speed analytics on time series data.

Solution Overview

This reference architecture consists of multiple interconnected components (Figure 1). These components include kdb+ deployed on a set of hosts that are connected to file storage on FlashBlade via a high performance ethernet network. This solution does not specify a set number of hosts or minimum network speeds, instead these design decisions will be made based on established needs and this reference architecture can be templatized without limits to any one component. Templatization also applies to FlashBlade product models and associated configurations and is further elaborated on in the Technology Overview section.

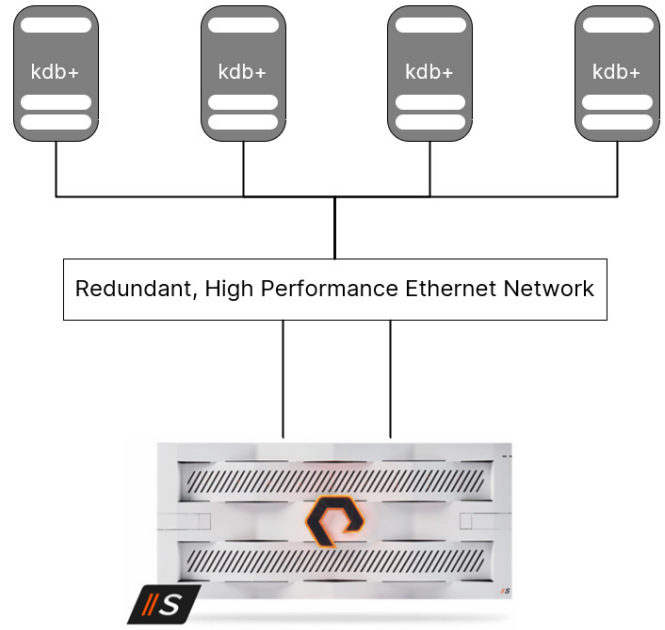


FIGURE 1 High-level overview of the interconnected components

Solution Benefits

This solution offers numerous benefits, significantly enhancing the performance, scalability, and efficiency of time-series analytics workloads. These features collectively create a robust, flexible, and cost-effective environment for managing high-performance data analytics, making it an ideal choice for financial institutions. The primary benefits include the following:

Unmatched performance: The proven performance for FlashBlade with the STAC-M3 audit demonstrates its exceptional capability to handle complex workloads, ensuring swift processing and analysis of critical datasets. This performance advantage is vital for maintaining a competitive edge in data-driven environments.

Scalable efficiency: Pure Storage solutions are tailored for highly parallel workloads, providing the seamless scalability necessary for improving analytic runtimes and developing complex financial models. This ensures that the system can grow alongside the increasing demands of the business. The solution allows for swift expansion of storage infrastructure in response to growing demands.

Maximize efficiency for rapid insights: Time-series databases require high throughput and low latency to support the rapid development, testing, and deployment of complex data models. FlashBlade ensures that trading strategies can move at market speed, accommodating the unprecedented pace of data growth.

Innovative scale-out performance and simplicity: The distributed, scale-out unstructured FlashBlade data platform provides the flexibility needed to process data at an exabyte scale. Its modular architecture allows for independent scaling of capacity and performance, ensuring the platform can meet evolving workload needs without disruption. Unlike traditional solutions that rely on complex clustered filesystems not optimized for this workload, FlashBlade offers a streamlined approach tailored for high-performance data processing

Cloud-agnostic deployment: FlashBlade can be implemented across any infrastructure supporting various protocols and tiers, offering flexibility and preventing vendor lock-in. This approach reduces reliance on specific cloud services and enhances system resilience, facilitating seamless operation across multiple platforms.

Optimized trading and analytics: Pure Storage delivers a modern data portfolio with simple manageability, proven persistence, massive scalability, and environmental sustainability. This empowers financial institutions to run mission-critical workflows with greater efficiency, and supports generating alpha through processing complex strategies with minimum latency and maximum scalability.

Technology Overview

This reference architecture is a set of connected technology components that communicate with one another over redundant high-speed ethernet connections (Figure 2). The hosts with kdb+ running access file storage on FlashBlade for the persistent storage of historical data.

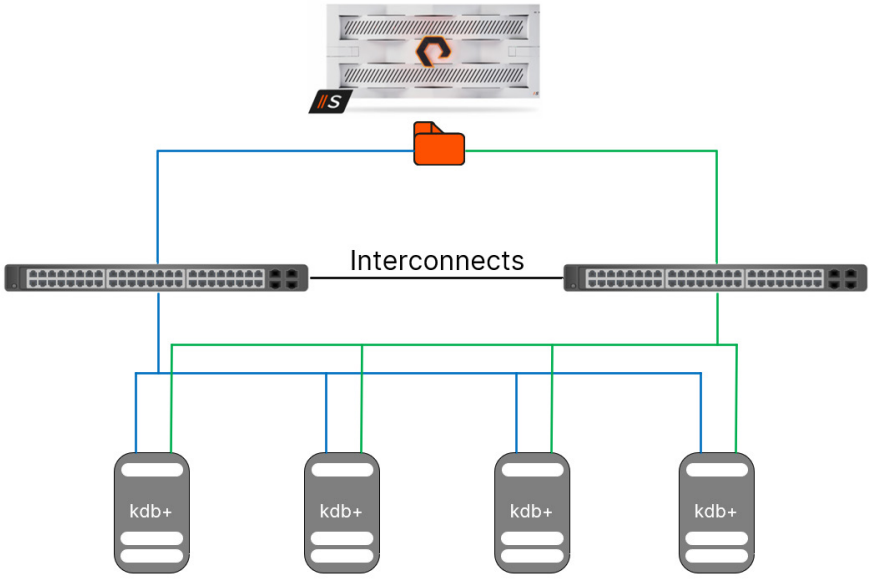


FIGURE 2 Technology component interconnects

Compute Resources

This reference architecture applies to physical multi-core servers with Linux based operating systems such as Red Hat Enterprise Linux, SUSE Enterprise Linux or Ubuntu, which are referred to as hosts. There are no minimums or maximums defined in this reference architecture, but for illustration purposes four (4) hosts are used.

Kdb+ implementations consist of several components, however as it is an in-memory database there is a separation between historical data (HDB) and current data (RDB). Figure 3 showcases the computer resources but lays out clear component ownership between the data types, real time data resides in memory within the hosts while historical data resides within persisted storage (see the section KX systems Kdb+ for more detail).

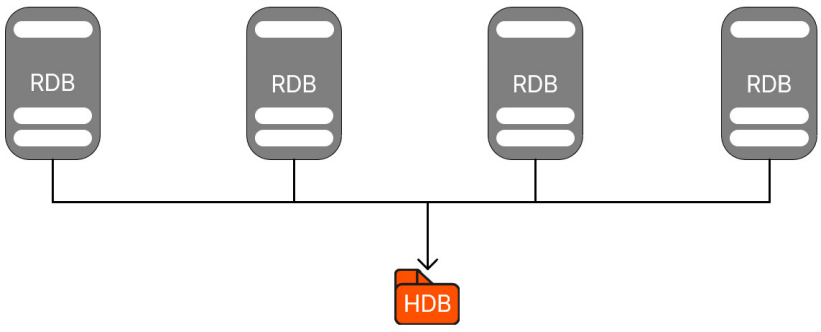


FIGURE 3 Compute resources

Network Resources

The network considerations for this reference architecture are the same as any enterprise IT infrastructure solution: availability, performance, and extensibility. The compute resources in the solution can attach to any compatible TCP/IPv4 or TCP/IPv6 network infrastructure with the general recommendation that minimum network speeds are capable of 25GbE. For this solution, the network should be configured using dual switches to eliminate a single point of failure.

Where possible client side (6-balance-alb) or switch side link aggregation (802.3ad) should be used to achieve the highest performance and availability.

Pure Storage FlashBlade

Pure Storage FlashBlade is tailored to meet the demands of modern data analytics requirements. Its innovative blade architecture prioritizes speed and efficiency, ensuring exceptional performance and scalability. This simplicity and adaptability empower organizations to effortlessly scale storage infrastructure to meet the demands of data-intensive workloads such as analytics, AI, and machine learning. With its all-flash design, FlashBlade consistently delivers high performance across various workloads, from backup to analytics and AI.

FlashBlade//S™ represents the evolution in enterprise scale-out storage, offering a blend of high density, capacity, performance, and scalability to meet the demands of modern applications.

Note: Relevant to this reference architecture is the modular approach FlashBlade//S takes to performance and capacity scaling. There are variable configuration options to both the number of blades in a chassis as well as the number of direct flash modules in each blade, allowing for greater optimization to right size capacity and performance needs in this platform.

FlashBlade//E™ is a cost-efficient data platform, aiming to provide effective performance, uncompromising reliability, and all-flash capabilities to a broader audience, especially those with budget considerations. As a cost-effective data solution within the FlashBlade product family, it provides exceptional value for analytics.

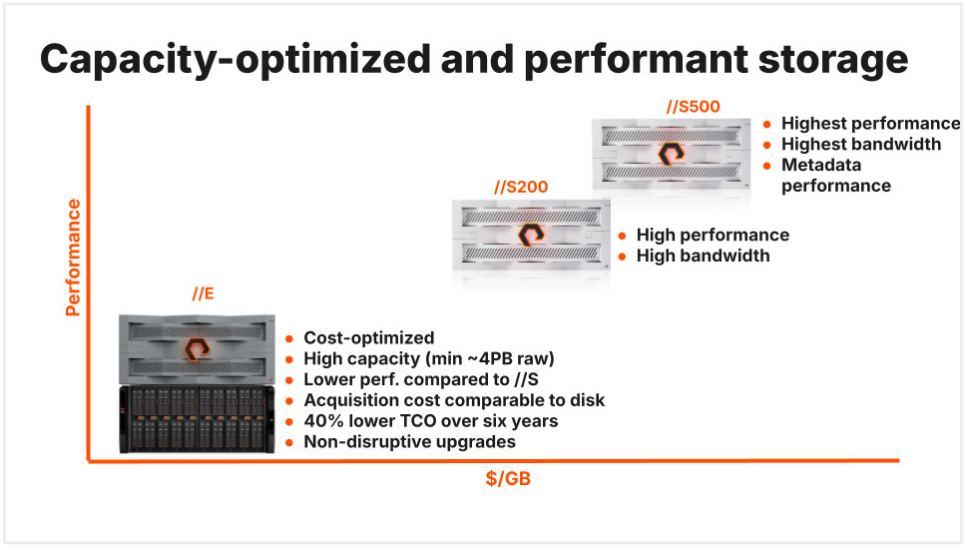


FIGURE 4 The FlashBlade family of products

Kx Systems kdb+

Kx Systems' kdb+ is a high-performance, column-store database designed for handling and analyzing large amounts of time-series data. Some key aspects of kdb+ are:

- **Time-series data**: kdb+ is optimized for time-series data, making it particularly well-suited for applications in financial services, such as trading and market data analysis, as well as other industries that require processing large datasets with time-based information.
- **Columnar storage**: Unlike traditional row-based databases, kdb+ uses a columnar storage format. This means that data is stored column-by-column rather than row-by-row, which allows for faster data retrieval and processing for certain types of queries, particularly those that involve large-scale aggregation or analysis.
- **q language**: kdb+ uses its own programming language called q, which is designed for querying and manipulating time-series data. q is a vector-based language, which means it can operate on entire arrays of data at once, providing significant performance advantages for certain types of operations.
- **High performance**: kdb+ is known for its high performance and low latency, making it a popular choice for applications that require real-time data analysis and decision-making.
- **Scalability**: kdb+ can handle large datasets and is scalable, making it suitable for enterprise-level applications. It can be deployed on a variety of hardware configurations, from single servers to distributed clusters.
- **Memory-mapped files**: kdb+ makes use of memory-mapped files to provide efficient data access and manipulation. This technique maps disk files into memory, allowing for faster read and write operations.
- **Applications**: While kdb+ is most used in financial services for tasks such as algorithmic trading, risk management, and market data analysis, it is also used in other industries such as telecommunications, utilities, and healthcare & life sciences for handling large volumes of time-series data.

Application use cases that use kdb+ are typically made up of multiple processes (Figure 5). These processes include the following components:

- **Data feed:** This is a source of real-time data; for example, financial quotes and trades from Bloomberg and LSEG or readings from a network of sensors.
- **Feed handler:** Parses data from the data feed to a format that can be ingested by kdb+. KX's Fusion interfaces connect kdb+ to a range of other technologies, such as R, Apache Kafka, Java, Python, and C.
- **Ticker plant:** Captures the initial data feed, writes it to the log file, and publishes these messages to any registered subscribers. Aims for zero-latency and includes ingesting data in batch mode. Manages subscriptions, adds and removes subscribers and sends subscriber table definitions. Handles end-of-day (EOD) processing.
- **Log file:** This is the file to which the Tickerplant logs the q messages it receives from the feedhandler. It is used for recovery: If the RDB must restart, the log file is replayed to return to the current state.
- **Real-time database (RTB):** Subscribes to messages from the Tickerplant, stores them in memory, and allows this data to be queried intraday. At the end of day it usually writes intraday data to the historical database (HDB) and sends it a new EOD message.
- **Real-time subscriber:** Subscribes to intraday messages and typically performs some additional function on receipt of new data–e.g., calculating an order book or maintaining a suitable with the latest price for each instrument.
- **Historical database:** Provides a queryable data store of historical data; for example, for creating customer reports on order execution times or sensor failure analyses. Large tables are usually stored on disk partitioned by date, with each column stored as its own file. The dates are referred to as partitions and this on-disk structure contributes to the high performance of kdb+.
- **Gateway:** The entry point into the kdb+ system. Responsible for routing incoming queries to the appropriate processes and returning their results. Can connect both the real-time and historical data to allow users to query across both. In some cases, a gateway will combine the result of a series of queries with different processes.

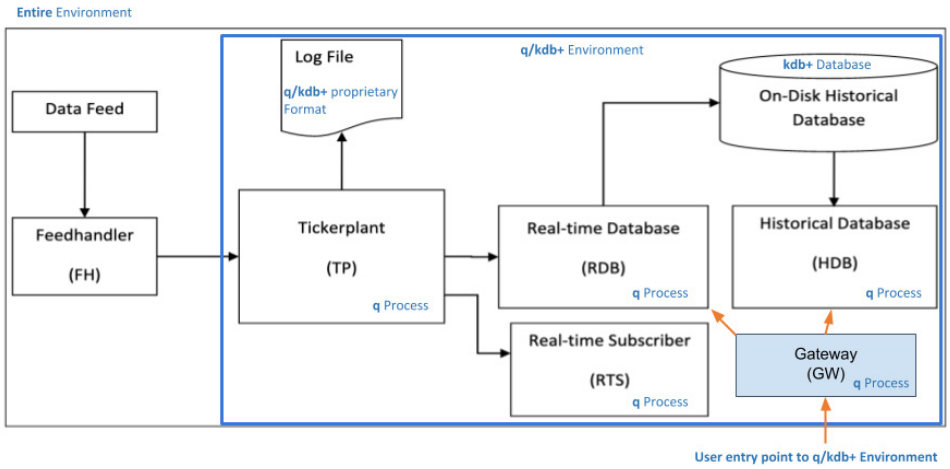


FIGURE 5 kdb+ process flow in a multi-process application architecture (image source Kdb+ and q documentation).

Technical Solution Design

The technical solution design in this reference architecture focuses on implementing a high-performance, cost-effective, and operationally seamless deployment of kdb+ for use cases such as high frequency trading, risk management and market research.

In this design, Kdb+ is deployed onto Linux-based bare metal hosts with network file system (NFS) mount points exported from a FlashBlade array, serving as high performance, persistent data layer for the HDB component of the kdb+ implementation as a segmented database. Data layer connectivity is performed over an ethernet network optimized for both performance and availability.

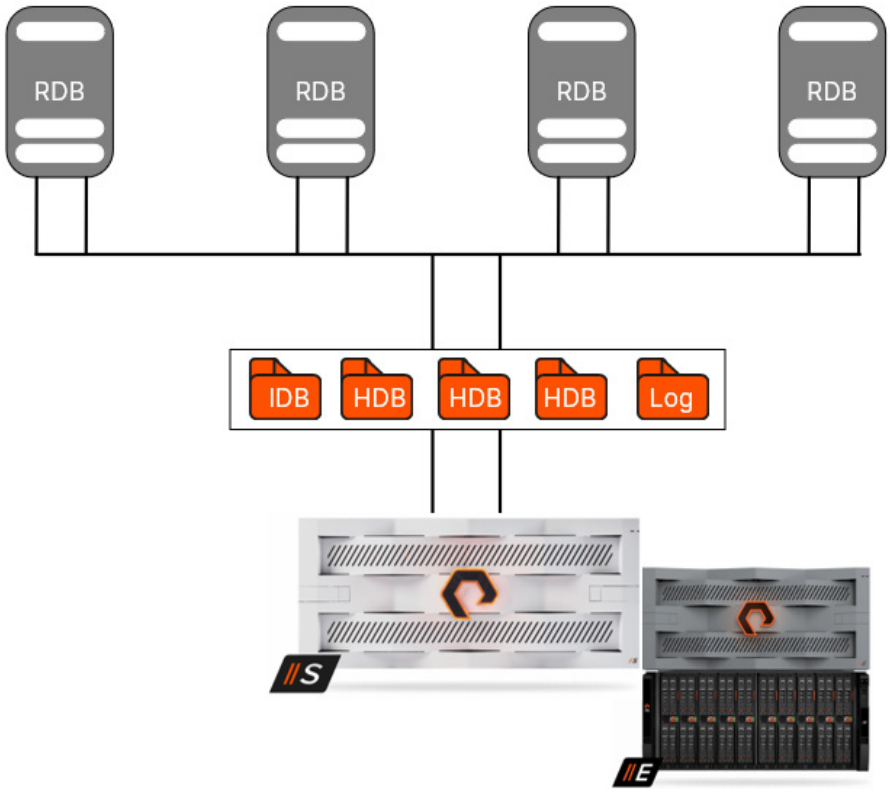


FIGURE 6 Reference architecture for kdb+ with FlashBlade.

Compute Layer

This reference architecture uses four (4) hosts in its design of the compute layer. The required servers are commodity x86-64 architectures with a recommended configuration for this reference architecture as follows:

- 16 cores
- The Kdb+ and q documentation specifies that at minimum it is recommended to have RAM of at least 4x expected data size, so for 5GB per day, the RDB machine should have at least 20GB RAM.
- Network cards/Ethernet adapter with at least 2×25Gbe ports

Note: These recommended configurations will vary based on customer need and are only intended to provide a starting template for the implementation journey.

Network Layer

Network ports on the hosts should be bonded in any one of the following configurations:

- Highest availability/Lowest performance—Active/Passive (Mode 1)
- Highest performance with no switch side configuration—Active Load Balancing (Mode 6)
- Highest performance and highest availability—802.3ad/LACP (Mode 4)

It is strongly recommended that dual switches are used and configured with an interconnect between them for availability and performance. Additional network optimizations for all interfaces are as follows:

- MTU—9000

For high performance use cases, the servers and storage should be connected to the same switches or have dedicated storage networking with no routing or hops between them.

FlashBlade networking is built using link aggregation groups, subnets virtual network interfaces (VIF):

- **Link aggregation groups:** Groups of physical uplinks that aggregate for availability and load balancing purposes.
- **Subnets**—A virtual representation of a network range and its associated configurations that is bonded to a link aggregation group. Includes options for VLANS and MTU settings.
- **Virtual interfaces:** An interface that clients/hosts connect to access specific storage protocols that are bonded to a subnet. Usually includes a dedicated static IP address and associated settings for VLAN and MTU options.

To avoid network bottlenecks this reference architecture recommends a link aggregation group capable of the same speed as the number of hosts x the speed of one network port on each. If the servers have 2 × 25Gb ports, then the FlashBlade link aggregation group needs to be capable of 100Gb minimum. All subnets and virtual interfaces should have the same MTU configuration.

Data Layer

FlashBlade provides file storage with the protocols SMB and NFS. This reference architecture focuses exclusively on the use of NFS with Linux hosts and is preferential to the use of NFSv3 for high performance scenarios.

File storage on FlashBlade uses storage object terminology such as File Systems and Object Stores. File Systems are management containers that can be exported via one or more storage protocols with the associated rules, policies, and configurations.

A minimum of one (1) file system should be exported using NFS from FlashBlade to all hosts, however depending on the database segmentation design more should be created.

Note: FlashBlade makes no distinction for performance or capacity when using one or more file systems, instead the advantage for segmentation is for the host operating system to utilize more threads per mount point. Individual filesystem capacity can be provisioned up to many petabytes.

The following mount point design and options are put forward as apart of this reference architecture:

- Mount point(s) - /mnt/kdb-data
- For segmented databases with multiple NFS exports, mount points numbering should be used e.g., /mnt/kdb-data01, /mnt/kdb-data02, /mnt/kdb-data03.
- Mount options
- **hard:** Sets the recovery behavior of the NFS client after an NFS request timeout.
- **nconnect=8:** Increases performance at scale by using more TCP connections between the client and the FlashBlade file service (should be at least 8 but can be increased). This is the only performance-enhancing configuration.
- **mountproto=tcp:** Specifies that the mount protocol to be used is TCP to ensure no data loss when using long running NFS connections.
- **Example:** 10.21.236.98:/kdb-data /mnt/kdb-data nfs rw,hard,fsc,nconnect=8, ,vers=3,tcp,timeo=600.

kdb+ Design Considerations

Designing a scalable architecture for kdb+ with FlashBlade involves several considerations. These considerations ensure high performance, scalability, resilience, and efficient data management to support the needs of the deployment.

Segmented Databases

As per the Kdb+ and q documentation, partitioned tables can be distributed across multiple storage devices to give them more space and support parallelization. This is achieved by providing multiple storage devices (see the Data Layer section) and then using a **par.txt** file to inform the database that it needs to unify the storage devices and present them as a single database for querying. This is an example of the formatting of a par.txt file:

```
/mnt/kdb-data01
/mnt/kdb-data02
/mnt/kdb-data03
```

The above par.txt results in the following directory design:

```
DISK 0       DISK 1           DISK 2
db         db             db
├── par.txt    ├── 2020.10.03      ├── 2020.10.04
└── sym        │   ├── quotes      │   ├── quotes
               │   ├── price       │   ├── price
               │   ├── sym         │   ├── sym
               │   └── time        │   └── time
               │   └── trades      │   └── trades
               │     ├── price     │     ├── price
               │     ├── sym       │     ├── sym
               │     ├── time      │     ├── time
               │     └── vol       │     └── vol
               ├── 2020.10.05      ├── 2020.10.06
               │   ├── quotes      │   ├── quotes
            ..           ..
```

Design Validation

This reference architecture is validated through an audited STAC-M3 report listed as SUT ID: KDB231122.

Note: The Securities Technology Analysis Center (STAC) offers technology research and testing tools grounded in standards created by the STAC Benchmark Council. This council is a consortium of over 300 financial institutions and more than 50 vendor organizations dedicated to developing benchmark standards beneficial for financial organizations. One of STAC's key offerings is STAC-M3, a set of industry-standard enterprise tick-analytics benchmarks for database software and hardware stacks that manage extensive time series of market data, commonly known as "tick data."STAC-M3 Benchmark details

Two STAC-M3 suites were tested: Antuco and Kanaga.

- The Antuco benchmark suite, with a size of 3.3TB, contains all the tick data for the year 2011. It is used for real-time performance testing with a varying number of concurrent users.
- The Kanaga benchmark suite, with a size of 60TB, includes all the tick data from 2003 to 2015. It represents historical tick data and is used for performance testing with large data sets and varying numbers of concurrent users.

STAC Provides additional details about the benchmark in the STAC-M3 benchmark overview.

Configuration

The following table lists out the component configuration used in the design validation:

Compute (Host) Configuration	**Compute nodes:** 8 x mid-range performance specifications, including dual Intel Xeon Platinum 8260 CPUs running at 2.40 GHz, 256 GiB of RAM, and dual port ConnectX-5 100GB network adapters. **Operating system:** Red Hat Enterprise Linux 8.6.
Storage Configuration	**Storage array:** A FlashBlade//S500 with 10 blades, 2×24TB (266 TiB total) direct flash modules per blade using Purity//FB 4.1.5. **Storage networking:** The FlashBlade is connected to the network with 8x network interfaces connected in a link aggregation group. **Storage provisioning:** 3 x NFSv3 Filesystems exported to all hosts with the only performance optimization being nconnect=8.
Networking	**Network hardware:** 2 × 100Gbe network switches are used to connect both hosts and storage. Both switches are configured as an MLAG which configures them to be presented as one logical switch for higher availability and performance. All network traffic between the hosts and storage is contained within them. **Network configuration:** A single VLAN is used for storage traffic between components.
KX Systems Kdb+	**Kdb+ version:** 4.0

Results

The primary metric in STAC-M3 benchmarking is query response time. Additionally, some I/O-focused benchmarks evaluate the rate of bytes read per second from persistent storage, excluding the server cache.

The performance of the STAC-M3 was compared against two distinct solutions:

- **Competing solution 1**: A setup comprising an equivalent number of database servers, three network-attached flash storage nodes, and an earlier version of the STAC Pack (KDB220506).
- **Competing solution 2**: A cloud-based configuration using a previous generation of the STAC Pack, alongside 15 database VMs accessing data across 40 VM instances (KDB210507).

In these comparisons, the STAC-M3 demonstrated superior performance in several areas:

- Out of 17 Antuco mean-response time benchmarks, 13 showed faster response times than the first solution.
- Out of 24 Kanaga mean-response time benchmarks, 17 showed faster response times than the first solution.
- Against the second solution, 9 out of 17 Antuco benchmarks and 12 out of 24 Kanaga benchmarks registered faster response times.

The following table highlight the major tests with findings of significance:

Test	Findings
High Concurrency Tests	**Description:** The STAC-M3 benchmark specifications marked as "v1" focus on storage system performance, with workloads deliberately heavy on I/O. These tests can submit requests from independent threads across multiple clients. For example, 10 clients each using 10 threads will create a total of 100 requesting threads to the storage system. A prominent v1 storage stress test from the STAC-M3 test suite is the VWAB-12DaysNoOverlap benchmark with 100 client threads. Each thread requests a 4-hour volume-weighted bid over 12 days for 1% of symbols, with no overlap in symbols for a date range of 1 to 5 years. **Finding:** FlashBlade//S500 effortlessly supports thousands of concurrent connections, far surpassing the demands of the STAC-M3 test suite. Notably, a speedup of 1.3 to 1.5 times was observed in 50-user, 12-day VWAB tests compared to competing solution 1.
Theoretical P&L	**Description:** The STAC-M3 benchmark rigorously evaluates queries for a basket of 100 trades on randomly selected dates. It determines future instances when 2X, 4X, and 20X the size of each trade is transacted per symbol, involving up to five days of forward analysis. The benchmark then computes the corresponding Volume Weighted Average Price (VWAP) and the total volume traded during those periods. **Finding:** Notably with FlashBlade//S500, a sevenfold (7x) increase in efficiency was observed in the 10-user theoretical Profit & Loss (P&L) analysis, compared to competing solution 1.
Market Snapshot	**Description:** The STAC-M3 benchmark evaluates queries for the most recent trade and quote data for 1% of symbols at a randomly selected time. **Finding:** Compared to competing solution 1, this approach demonstrated a 5.8-fold increase in efficiency for a 10-user market snapshot. It also achieved an 11-fold speedup compared to competing solution 2.
Volume Curves	**Description:** The STAC-M3 benchmark creates an average volume curve (using minute intervals aligned on minute boundaries) for 10% of symbols over 20 randomly selected days. **Finding:** An 8.4x speedup in the 10-user volume curve was noticed when compared to competing solution 2.
Year High Bid	**Description:** The STAC-M3 benchmark efficiently determines the maximum bid over the year for 1% of symbols. **Finding:** Using the Kanaga suite, a speedup of 1.2 to 1.4 times was observed in the 1-user annual high bid analysis, compared to competing solution 1.
Write Test	**Description:** The STAC-M3 benchmark includes a single write test for the Antuco suite. **Finding:** Performance was noted to be strong. Additionally, the latency observed on the FlashBlade//S500 was exceptionally low, further emphasizing its scalability.
Storage Efficiency	**Description:** The Antuco suite assesses storage efficiency. **Finding:** FlashBlade reported an efficiency of 149%. Furthermore, FlashBlade//S500 demonstrated data reduction capabilities, achieving a 2.1:1 ratio for the STAC-M3 Antuco dataset. This level of data reduction is particularly noteworthy for large datasets, as it results in substantial space savings. Consequently, it significantly reduces overall storage costs over time.

Deployment

This section provides detailed guidance for deploying kdb+ on FlashBlade file storage, including installation, and configuration of key components.

FlashBlade

This deployment guidance covers the core steps in storage provisioning for kdb+. It assumes the array has been deployed but that specific configuration for the technology components used for this reference architecture has not been performed.

FlashBlade Network Provisioning

Follow the below steps to provision the network on FlashBlade for the kdb+ implementation.

1. In the FlashBlade graphical user interface navigate to the **Settings** section and select the section for **Link Aggregation Groups**. A link aggregation group should already be provisioned as a part of any FlashBlade field deployment. In the below example, the uplink group already exists and will be used for future network configuration.

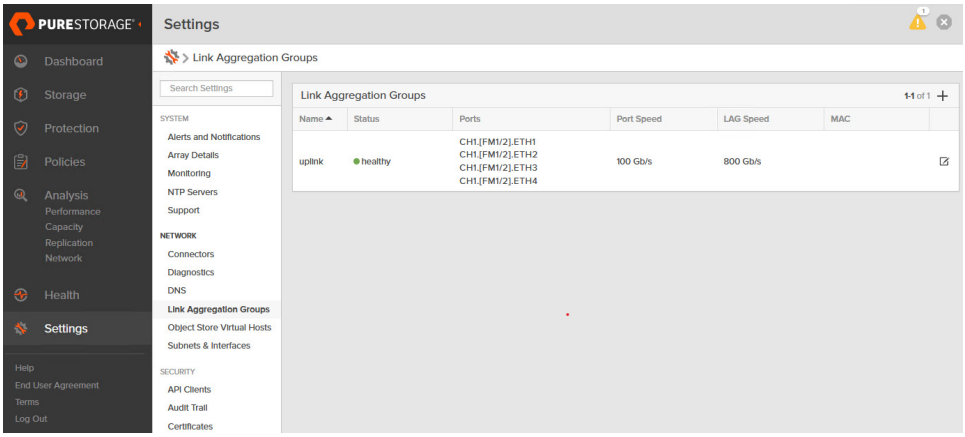


FIGURE 7 Link Aggregation Groups in FlashBlade settings

2. Navigate to the **Subnets and Interfaces** section, identify the **Subnets** view, and select the **+** in the top right-hand corner to create a new subnet.

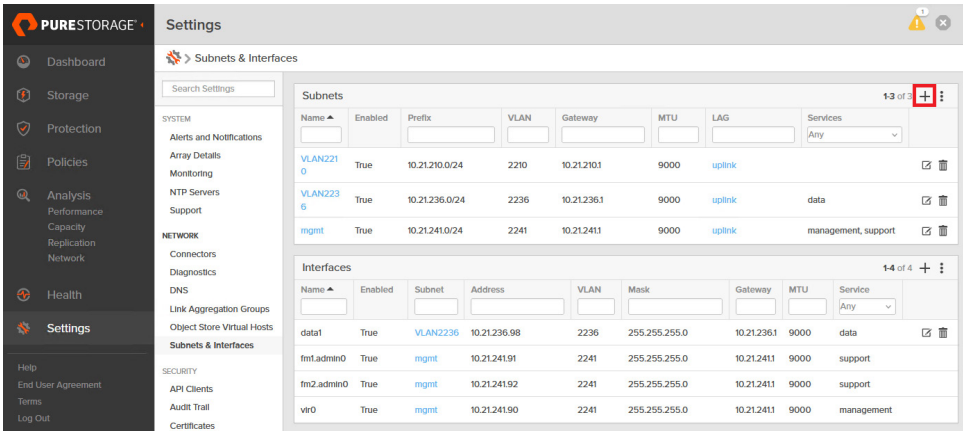


FIGURE 8 The Subnets and Interfaces section, focusing on Subnets.

3. In the **Create Subnet** prompt which appears, provide a **name**, **network prefix**, a **VLAN**, a **gateway** for the network, the **MTU** and select the **Link Aggregation Group (LAG)** to be used then select **Create**.

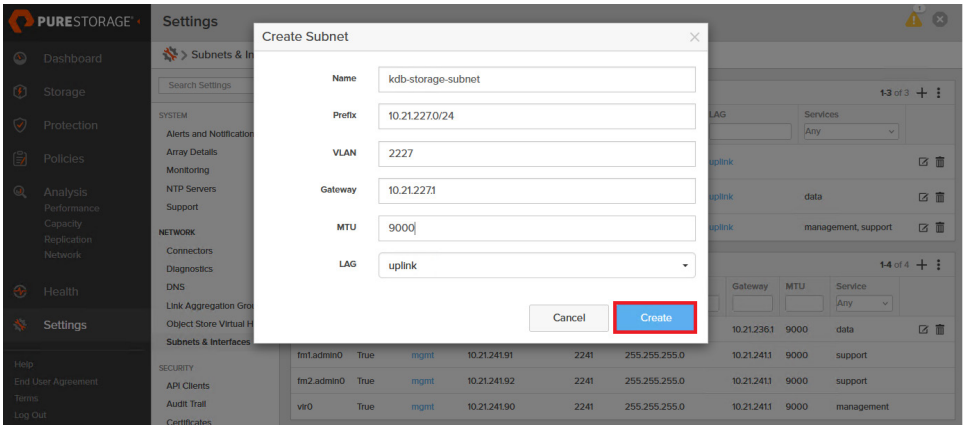


FIGURE 9 The Create Subnet prompt.

4. Once the subnet has been created identify the **Interfaces** view and select the **+** in the top right-hand corner to create a new interface.

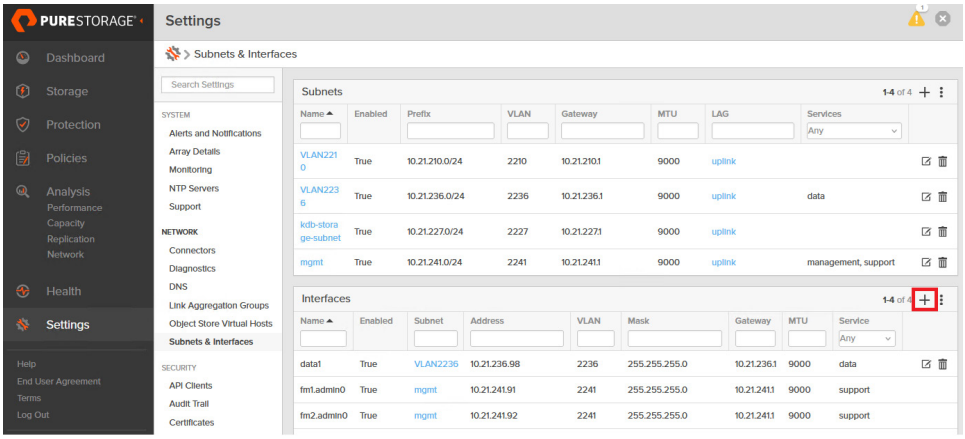


FIGURE 10 The Subnets and Interfaces section, focusing on interfaces

5. In the **Create Network Interface** prompt which appears enter a value for the **name**, select the **subnet** defined in step 4, assign a static **IP address**, and then select the **data** service type. Once complete select **Create**.

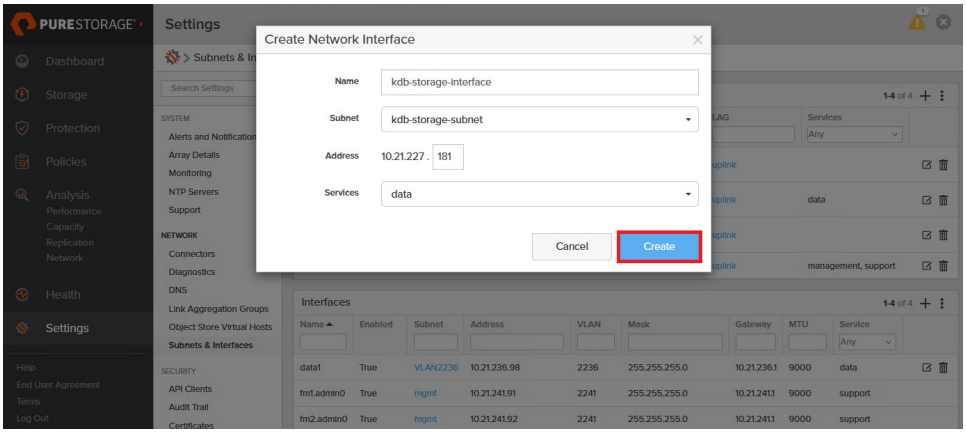


FIGURE 11 The Create Network Interface

6. Review that the Interface has been successfully created and that all of its properties are correct.

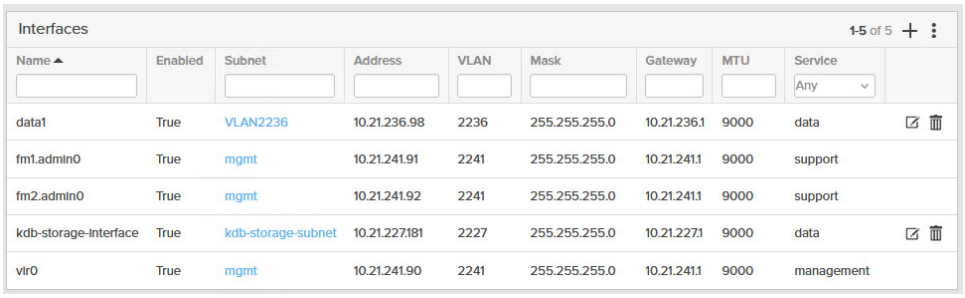


FIGURE 12 The list of interfaces with the new interface created.

Provisioning NFS File Systems on FlashBlade

Follow the below steps to provision the FlashBlade file systems for the kdb+ implementation.

1. In the FlashBlade graphical user interface navigate to the **Storage** section and Select the **File Systems** tab. In this tab identify the section for **File Systems** and select the **+** to create a new file system.

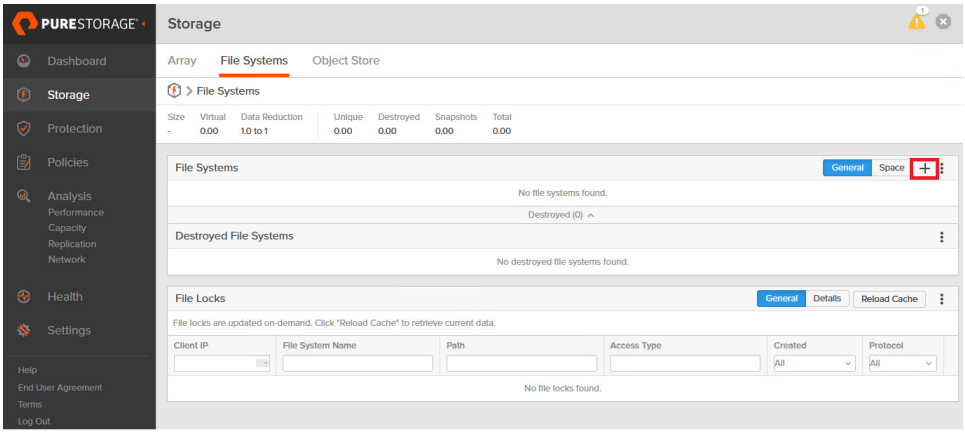


FIGURE 13 The File Systems view in the Storage section of the FlashBlade graphical user interface.

2. (Repeat this step for each file system that needs to be created). Within the **Create File System** dialog, enter in a **name** and **size**, select the **NFSv3** check box and specify any **use rules** required (leaving the use rules blank results in an any user/any host policy being applied). Once complete select **Create**.

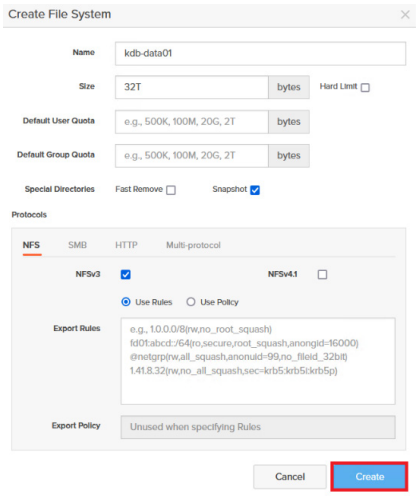
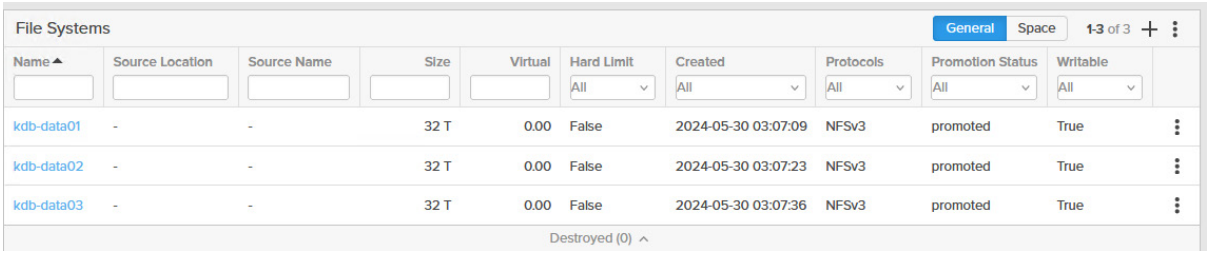


FIGURE 14 The Create File System dialog.

3. Once complete, review the file systems present within the File Systems view.



Host Operating System (RHEL, Centos, SUSE)

Follow the below steps on all hosts to provision the host operating system and ensure that all requirements are in place for the kdb+ implementation.

1. Set the read ahead value for NFS mounts, outcomes may vary dependent on the client side workload—adjust the read_ahead_kb value to suit, by creating a new file called **99-pure-nfs.rules** in **/etc/udev/rules.d/** with the following contents

```
SUBSYSTEM=="bdi", ACTION=="add", PROGRAM="/bin/awk -v bdi=$kernel 'BEGIN{ret=1} {if ($4 == bdi) {ret=0}}
END{exit ret}' /proc/fs/nfsfs/volumes", ATTR{read_ahead_kb}="2480"
```

2. Set the CPU power governor to performance mode using the **cpupower** command:

```
cpupower frequency-set -g performance
```

3. Create the directories for the NFS mount points using the **mkdir** command:

```
mkdir -p /kdb/kdb-data01
mkdir -p /kdb/kdb-data02
mkdir -p /kdb/kdb-data03
```

4. Check what NFS exports can be seen on the previously created FlashBlade interface using the **showmount** command.

```
# showmount -e 10.21.227.181
Export list for 10.21.227.181:
/kdb-data01 *
/kdb-data02 *
/kdb-data03 *
```

5. Edit /etc/fstab and append the following entries to ensure persistent mounting of the NFS shares with the correct connection options.

```
10.21.227.181:/kdb-data01    /kdb/kdb-data01 nfs   nconnect=8,mountproto=tcp,hard 0 0
10.21.227.181:/kdb-data02    /kdb/kdb-data02 nfs   nconnect=8,mountproto=tcp,hard 0 0
10.21.227.181:/kdb-data03    /kdb/kdb-data03 nfs   nconnect=8,mountproto=tcp,hard 0 0
```

6. Reload systemd and mount the NFS shares using the mount command:

```
# systemctl daemon reload
# mount -a
```

7. Check the mount points are correct using the **df -h** command.

```
# df -h
……….
10.21.227.181:/kdb-data01  32T   0  32T  0% /kdb/kdb-data01
10.21.227.181:/kdb-data02  32T   0  32T  0% /kdb/kdb-data02
10.21.227.181:/kdb-data03  32T   0  32T  0% /kdb/kdb-data03
……
```

KX Systems Kdb+

The kdb and q documentation provides deeper details on how to obtain and deploy kdb+. The following steps are only examples of steps that may need to be taken to optimize the deployment for previously configured storage options.

1. For a segmented database edit the **db/par.txt** file and ensure it has the following entries:

```
/kdb/kdb-data01
/kdb/kdb-data02
/kdb/kdb-data03
```

2. Check that all the hostnames of nodes used in the implementation are present in the config/nodes.txt file:

```
DB-05
DB-06
DB-07
DB-08
```

Conclusion

This reference architecture with KX Systems kdb+ on FlashBlade offers a comprehensive solution tailored for the demanding environments of financial institutions. By integrating Pure Storage FlashBlade with KX Systems' kdb+, this architecture significantly enhances the performance and scalability of tick data analytics, enabling faster and more accurate data processing.

This solution provides unmatched performance validated by the STAC-M3 benchmark, ensuring swift processing of complex workloads, which is crucial for maintaining a competitive edge in high-frequency trading and market research. Its scalable efficiency allows seamless growth to meet increasing data volumes and analytical complexities, supporting the rapid development and deployment of sophisticated financial models.

Furthermore, the architecture ensures robust data integrity and high availability, minimizing disruptions and ensuring consistent operations critical for financial markets. The flexibility of cloud-agnostic deployment and the innovative scale-out performance of FlashBlade provide a resilient and adaptable infrastructure capable of evolving with the needs of the business.

In conclusion, the combination of kdb+ and FlashBlade offers financial institutions a powerful, flexible, and cost-effective environment for managing high-performance data analytics. This reference architecture not only addresses the current challenges of tick data analytics but also positions organizations to efficiently handle future data growth and computational demands, empowering them to make informed, timely decisions and maintain a competitive advantage in the market.

Additional Resources

- Solution Brief: Unlocking High-performance Time Series Analytics
- White Paper: Quantitative Trading with Pure Storage Solutions
- Solution Brief: Pure Storage Solution for Quant and HFT
- Technical White Paper: Pure Storage GenAI RAG Platform for Financial Services with NVIDIA NeMo Microservices and KX KDB.AI
- Blog: Pure Storage STAC-M3 Benchmark Testing Results for High-performance and Quant Trading

PS2653-01-en 08/24