WHITE PAPER

MAXIMIZING REDIS ENTERPRISE PERFORMANCE WITH KUBERNETES ON THE NVME-EQUIPPED DIAMANTI BARE-METAL CONTAINER PLATFORM

For critical enterprise applications such as databases, containers have unlocked enormous potential in terms of workload performance, as well as speed of deployment and agility. Naturally, the choice of database platform and supporting infrastructure play a major role in realizing that potential.

This paper focuses on how Diamanti's purpose-built bare-metal container platform running Kubernetes enhances performance for Redis on Flash (RoF), a leading in-memory database implementation.

ADVANTAGES OF REDIS LABS' CONTAINERIZED IN-MEMORY DATABASES

An in-memory database like Redis Enterprise accelerates data access from outside an application's main memory space. By storing the results of database calls, frequent JavaScript calls, application session information, and other ephemeral computed data, an application can perform faster and be more responsive to user actions as a consequence of sharing application results between requests. This fast and intelligent response to user actions, within milliseconds, keeps users engaged, at any volume and scale.

LEVERAGING NVME

Typically, in-memory databases consume system RAM. Therefore, scaling the database involves expanding system RAM, and then adding server nodes. Here, cost and data center footprint will increase significantly.

Redis Enterprise offers a more efficient alternative with Redis on Flash (RoF). RoF enables Redis databases to span both RAM and dedicated flash memory. While keys are always stored in RAM, RoF intelligently manages the location of their values (RAM vs Flash) in the database via a LRU-based (least-recently-used) mechanism. Hot values will be in RAM, but less frequently-used (warm) values will be ejected to flash memory. This allows for much larger datasets with RAM-like latency and performance, but at dramatically lower cost than an all-RAM database.

REDIS ON FLASH: BENEFITTING FROM CONTAINERIZATION

Redis on Flash benefits substantially from containers in terms of deployment speed, agility, and scalability. Building a database as a set of smaller composable elements (as opposed to a larger monolithic entity) significantly enhances ease and speed of development, testing, and deployment. RoF database instances can be rapidly spun up and ported across Kubernetes environments.

ADVANTAGES OF THE DIAMANTI BARE-METAL CONTAINER PLATFORM

Diamanti's bare-metal container platform is a hyper-converged container solution for Kubernetes. The platform comprises a highly scalable Kubernetes stack (including unmodified versions of both Kubernetes and Docker) that can be deployed and configured in a fraction of the time it would take IT Operations teams to stand up infrastructure built from legacy storage, networking, and software.

The Diamanti platform is built on x86 architecture without requiring a hypervisor, and features proprietary network and storage controllers that allow for direct integration with existing data center networks and self-service storage management for stateful containers. Using industry standard SR-IOV for virtualizing network and storage interfaces, each container deployed on the Diamant platform runs with dedicated network throughput and IOPS, thereby guaranteeing quality-of-service (QoS) for any application. Ultimately, dedicated network and storage performance queues ensure high container density and deliver substantially better performance and resource utilization over legacy infrastructure.

DIAMANTI BARE-METAL CONTAINER PLATFORM: OPTIMIZED FOR CONTAINERIZED DATABASES

Early adopters of containers initially used them to run applications which didn't require persistent storage. CPU and memory were easy, portable concepts between different hardware platforms. However, storage for stateful applications was a difficult challenge, given that local storage was usually not sufficient, and that performance couldn't be guaranteed due to being shared with other application containers.

As more and more organizations sought to containerize legacy databases and build new ones based on microservices architecture, persistent storage has become an increasingly important requirement. In light of this, Diamanti made a key contribution to the open-source community with its development of the FlexVolume plug-in, which allows a standard way of integrating 3rd party storage into Kubernetes. Now, all types of developers and architects have storage that is accessible and configurable from the Kubernetes API and PodSpecs.

The Diamanti platform offers a turnkey solution for container storage management, and delivers container-granular quality-of-service (QoS). Each Diamanti node offers up to 6.4TB of NVMe flash storage, which Redis on Flash databases can leverage for higher performance.

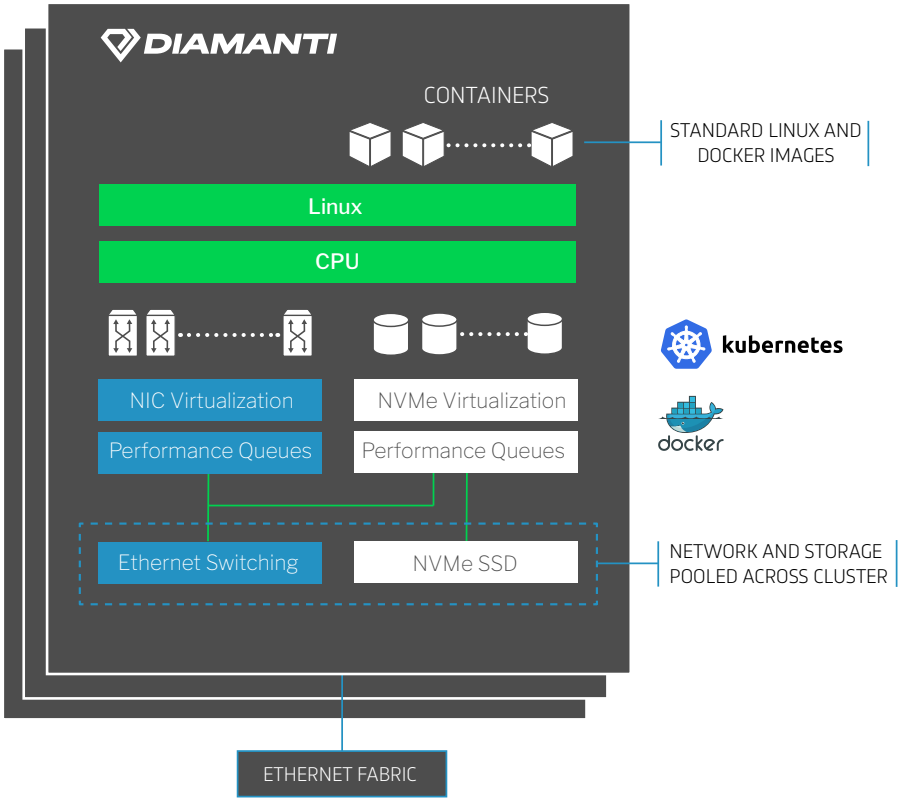


Figure 1: Diamanti bare-metal node

RESOURCE USAGE ON BARE-METAL CONTAINER INFRASTRUCTURE

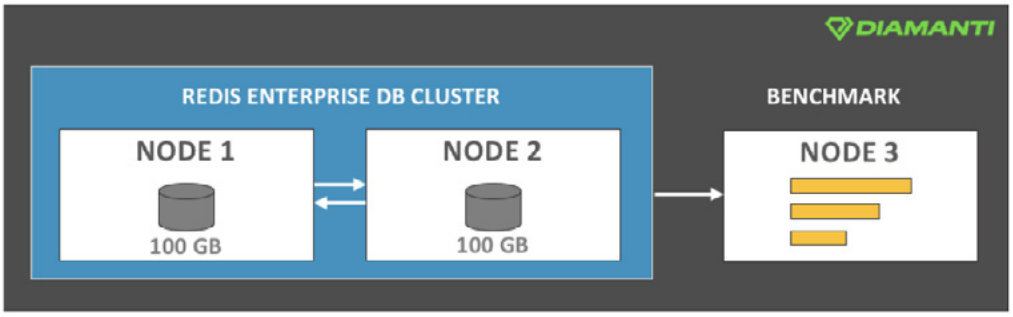
Running database applications in containers allows for clusters to be rapidly deployed, scaled, and decommissioned on-the-fly. Herein lies the primary difference between a traditional, VM-based container stack and a bare-metal container stack such as the Diamanti platform. With VM-based infrastructure, there is considerable CPU and memory overhead due to over-provisioning, leaving a non-trivial amount of system resources unused by the actual application. Diamanti's platform maximizes compute resources for running applications by stripping the container stack of expensive and resource-intensive hypervisor and VM layers, allowing applications to use upwards of 90% of hardware resources with dedicated storage and network traffic. Furthermore, the Diamanti platform natively features NVMe flash storage which is leveraged by Redis Enterprise for low latency, high performance even as database instances are scaled dynamically across growing clusters.

TESTING REDIS ENTERPRISE DATABASE ON DIAMANTI

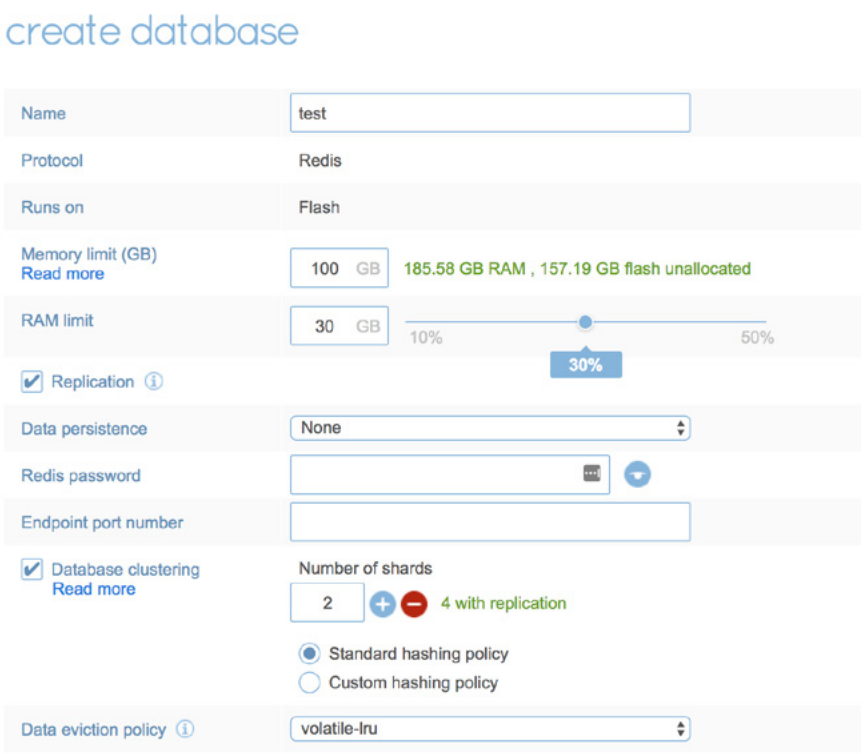
A three-node Diamanti cluster was used to test Redis Enterprise version 5.0.0-31 with the following specifications for each node:

Diamanti D10 Three-Node Cluster	
Network	4x10 GbE VNIC (Virtual Network Controllers) per node
Data Storage	3.2 TB NVMe flash storage, per node (9.6 TB total)
Compute	CPU: 2x E5-2630V4 2.2 GHz Intel® Xeon® Processors, per node RAM: 128 GB, per node
Container Runtime	Docker version 1.12.6
Orchestration	Kubernetes (Kubernetes 1.8 certified)

The three-node Diamanti cluster was configured as follows:



The Redis Enterprise Instances were created using the following configurations:



DATABASE INITIALIZATION AND BENCHMARKING PROCESS

STEP 1

Populate the database with values. The following command fills the database with 75 million entries, each with a 500-byte payload:

```
memtier_benchmark -s your-nodes-fully-qualified-name-or-ip-endpoint
-p your-endpoint-port --hide-histogram --key-maximum=75000000 -n
allkeys -d 500 --key-pattern=P:P --ratio=1:0
```

STEP 2

Centralize the database with approximately 21 million records which will reside in RAM:

```
memtier_benchmark -s your-nodes-fully-qualified-name-or-ip-endpoint
-p your-endpoint-port --hide-histogram --key-minimum=27250000
--key-maximum=47750000 -n allkeys --key-pattern=P:P --ratio=0:1
```

STEP 3

Run the benchmark against the primed database:

```
memtier_benchmark -s your-nodes-fully-qualified-name-or-ip-
endpoint -p your-endpoint-port  --pipeline=11 -c 20 -t 1 -d 500
--key-maximum=75000000 --key-pattern=G:G --key-stddev=5125000
--ratio=1:1 --distinct-client-seed --randomize --test-time=600
--run-count=1 --out-file=test.out
```

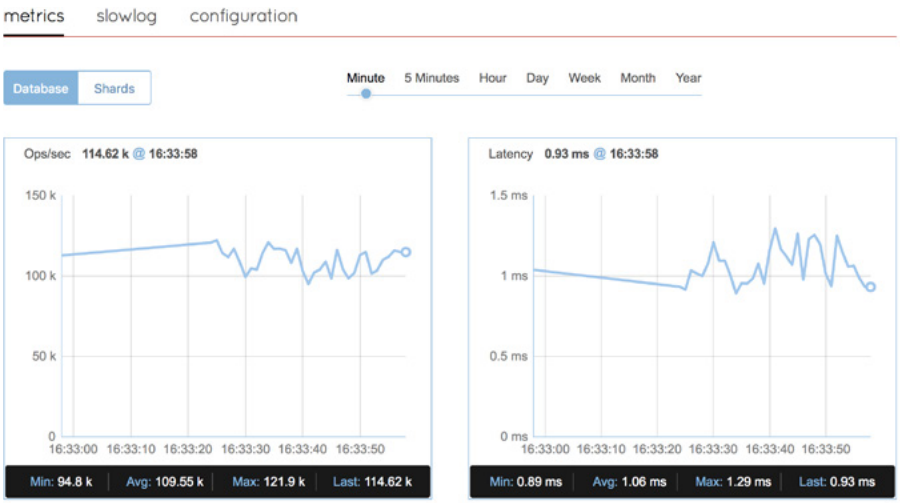
DATABASE PERFORMANCE TEST RESULTS

On the three-node Diamanti cluster, we initialized the test database as described in the previous section. Screenshots in this section reflect the performance results for the Diamanti platform as viewed from the Redis console.

This test illustrates that an average throughput of more than 100,000 ops/sec at a sub-millisecond latency can be achieved with only four shards running on a three-node cluster with two serving nodes, which meets the minimum performance requirements of typical real world database use cases. In the event more throughput or memory is needed, you can scale your cluster by simply adding more shards and nodes.

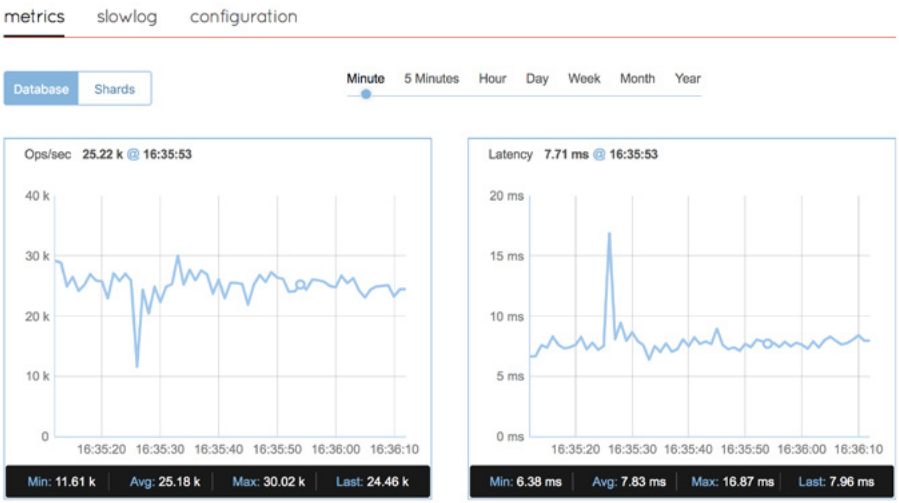
STEP 1: PERFORMANCE RESULTS DATABASE POPULATION WITH KEY/VALUE PAIRS

Running on the Diamanti platform, Redis Enterprise achieved an average of 110k operations/sec with sub-millisecond latency between database operations.



STEP 2: PERFORMANCE RESULTS—DATABASE PRIMING

In priming the database, Redis Enterprise performed at an average of 25k operations/sec and 8 ms of latency between operations.



STEP 3: PERFORMANCE RESULTS—DATABASE BENCHMARK

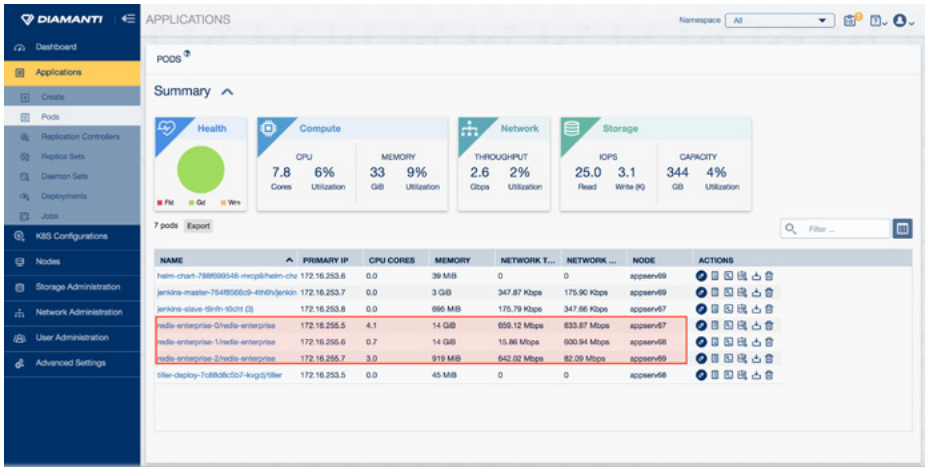
In running the database benchmark test on the Diamanti bare-metal container platform, the Redis Enterprise database scored an average of 130k operations/sec with sub-millisecond latency. Diamanti achieved 13% higher database performance than a traditional VM-based container stack.

DATABASE RESOURCE CONSUMPTION ON THE DIAMANTI BARE-METAL CONTAINER PLATFORM

Across each of the three test stages, we examined how the Diamanti platform's resources were used.

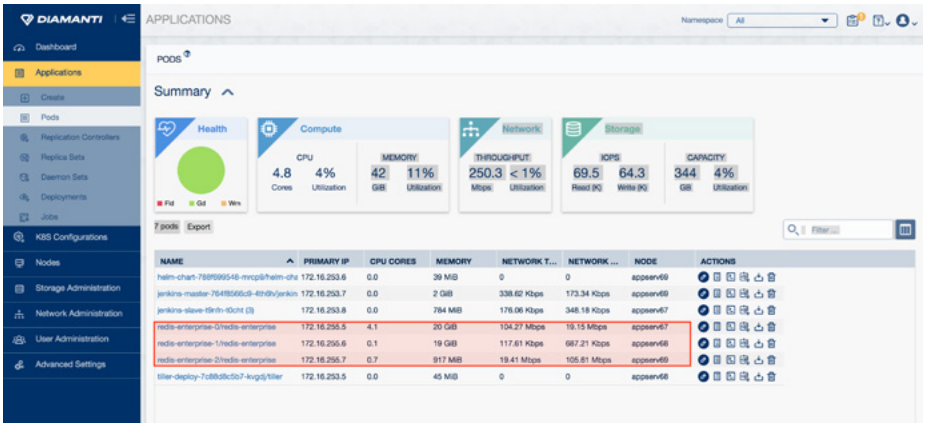
STEP 1: DATABASE POPULATION

During database population, the three-node Diamanti cluster pushed an average of 660 Mbps input/output from node 2 (benchmark) to node 0 (primary database). The primary database and its values were then replicated to node 1, for a combined 2.4 Gbps of traffic generated within the cluster.



STEP 2: DATABASE PRIMING

During priming, the tooling attempts to randomize the database by expiring, deleting, adding, and preparing the database contents to approximate a real-world stress test. For this process, the Diamanti cluster sustained 170 Mbps of traffic throughput between the nodes and less than 1% utilization of the overall capacity of the cluster. Also, each node is using approximately 20 GB of memory in order to perform the requested database transactions—equivalent to only 9-10% utilization of the Diamanti cluster's pooled memory.

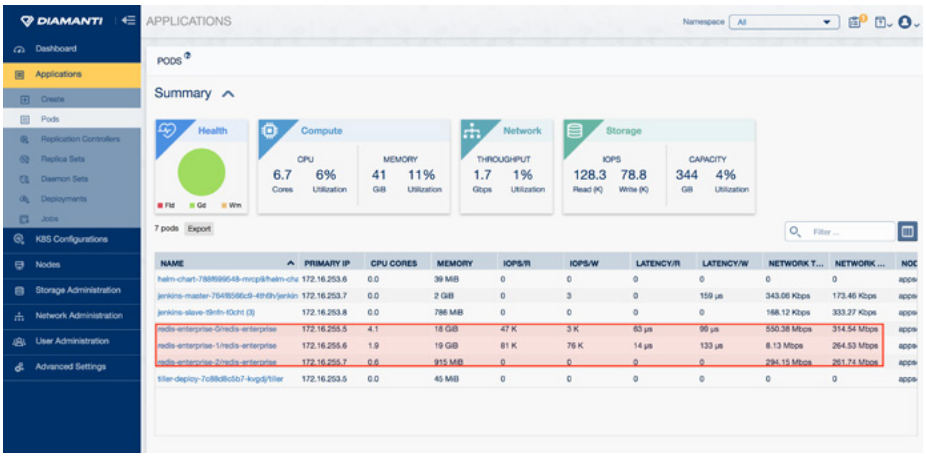


STEP 3: BENCHMARKING

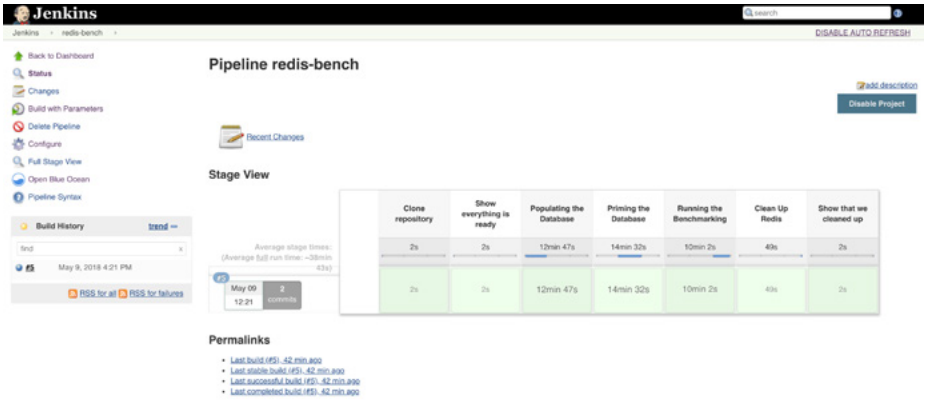
During database benchmarking, the Diamanti dashboard shows combined input/output network traffic between database nodes of almost 1.8 Gbps. This translates to less than 2% overall utilization of the Diamanti cluster's storage capacity.

As expected, CPU consumption rises as more database requests are made. Still, only between 6% and 7% of available CPU cycles are being consumed.

What is most noteworthy, however, is the read and write latencies achieved by the Diamanti cluster. This sub-millisecond latency is a direct result of Diamanti's use of all-flash NVMe storage coupled with the pre-integrated and optimized Kubernetes infrastructure leveraged extensively by the Redis Enterprise instances.



The Jenkins CI/CD pipeline was used to run the benchmark process in order to determine exactly how long each step ran for in the database performance test. It's also worth noting that the performance numbers achieved were actually sustained over longer periods of time, and were not simply intermittent performance spikes.



Finally, had we gone ahead and altered the data eviction policy of the database, or changed the threading of the benchmarking tool, we could have achieved even better performance.

SUMMARY

A reliable, performant in-memory database platform is an important enterprise tool for transactional, analytical, and hybrid workloads. Therefore, it is crucial to run them on infrastructure that maximizes their performance while optimizing costs.

In testing Redis Enterprise, with its support for NVMe flash storage, the results concluded that Diamanti's bare-metal container platform offers 13% better overall performance with zero tuning required, compared to container stacks built with legacy infrastructure.

Applications take advantage of accelerated storage access, both local to an appliance and across the cluster, to achieve consistent performance and to avoid bottlenecks from software overlays. As a result, Diamanti's converged all-flash storage, overlay-less networking, powerful compute, and certified Kubernetes architecture significantly speeds time-to-deployment and reduces both datacenter footprint and operational costs, compared to other types of on-premises infrastructure approaches.

Some of Redis Labs' recent findings also conclude that the use of flash is a major factor in contributing to efficiency:

> *'If you wanted to build a RAM-only version of this database, you'd need three times the number of nodes. Which, in turn, would translate to three times the infrastructure cost for running the database. Using Flash as a RAM extension provides quite the savings."*

In terms of performance and efficiency, enterprises are getting the best of both worlds with Diamanti; automated layer-2 networking and persistent container storage that enable dedicated performance queues containerized databases require, along with scalable, turnkey bare-metal container infrastructure that users can operationalize in 15 minutes.

ABOUT DIAMANTI

Diamanti's bare-metal container platform gives infrastructure architects, IT operations, and application owners the speed, simplicity, efficiency, and control they need to run stateful containerized applications at scale. With open-source Docker and Kubernetes fully integrated, together with purpose-built hardware and complete support for the entire stack, the Diamanti platform is a proven full container stack that deploys in minutes.

ABOUT REDIS LABS

Modern businesses depend on the power of real-time data. With Redis Labs, organizations deliver instant experiences in a highly reliable and scalable manner. Redis Labs is the home of Redis, the world's most popular in-memory database, and commercial provider of Redis Enterprise that delivers superior performance, matchless reliability and unparalleled flexibility for personalization, machine learning, IoT, search, ecommerce, social and metering solutions worldwide.