Intel® Rack Scale Architecture Utilizing EMC® ScaleIO® and CoprHD Open Source Storage Controller

Whitepaper

August 2015

Revision 001
## Contents

1. **Introduction** .......................................................................................................................... 5  
   1.1 Executive summary ............................................................................................................... 5  
   1.2 Terminology ....................................................................................................................... 5  
2. **Overview** ............................................................................................................................. 6  
3. **The EMC® ScaleIO® Storage Personality** ............................................................................ 7  
4. **CoprHD** .................................................................................................................................. 9  
5. **Intel® Rack Scale Architecture** ............................................................................................ 11  
6. **Intel® Rack Scale Architecture with EMC® ScaleIO® and CoprHD Demonstration** ............ 13  
   6.1 EMC® ScaleIO® .................................................................................................................. 18  
   6.2 CoprHD ................................................................................................................................ 19  
7. **Future Directions** ............................................................................................................... 26

## Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>EMC® ScaleIO® network diagram</td>
<td>7</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Management O&amp;M tools</td>
<td>9</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Basic Intel® Rack Scale Architecture benefits</td>
<td>11</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Intel® Rack Scale Architecture overview</td>
<td>12</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Bulldog Creek SDV compute tray</td>
<td>13</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Orchestration layers</td>
<td>14</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Process discovery flow</td>
<td>15</td>
</tr>
<tr>
<td>Figure 8</td>
<td>OpenStack rack resource inventory</td>
<td>16</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Sample JSON script sent with hardware description</td>
<td>16</td>
</tr>
<tr>
<td>Figure 10</td>
<td>OpenStack node assembly page</td>
<td>17</td>
</tr>
<tr>
<td>Figure 11</td>
<td>ScaleIO® user interface</td>
<td>18</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Add ScaleIO® to physical assets</td>
<td>19</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Add a virtual array</td>
<td>20</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Add a virtual array</td>
<td>21</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Add a project</td>
<td>21</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Service catalog</td>
<td>22</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Select block storage service</td>
<td>23</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Create block volume</td>
<td>23</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Block volume creation options</td>
<td>24</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Resources</td>
<td>25</td>
</tr>
</tbody>
</table>
## Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>Initial public release.</td>
<td>August 17, 2015</td>
</tr>
</tbody>
</table>
1 Introduction

This document contains information about the Intel® Rack Scale Architecture Pod Manager API Specification (332869), which was designed and implemented for the Intel® Rack Scale Architecture Software v1.1 release for the Bulldog Creek SDV. This document is intended for designers and engineers working with Intel® Rack Scale Architecture software. Objectives include the following:

- Introduce the Intel® Rack Scale Architecture storage concept.
- Provide overviews of CoprHD and EMC® ScaleIO® benefits.
- Demonstrate CoprHD and ScaleIO® on Intel® Rack Scale Architecture.
- Preview what’s next.

1.1 Executive summary

Intel® Rack Scale Architecture provides an evolving reference design that allows cloud datacenter operators to catalog and logically pool compute, network, and storage resources. These pools of resources contain memory, CPU, storage media, and other physical attributes of the individual nodes and racks, and enable higher-level orchestrators. The pools use application programming interfaces (APIs) to match these resources with application-centric requirements. Software-defined storage (SDS) platforms (for example, EMC® ScaleIO® and open source alternatives) and SDS controllers (for example, the open source CoprHD project) provide the methods required to compose various disparate resources into resilient systems that can then be provisioned, matched, and managed according to application-based service level objectives (SLOs).

Intel® Rack Scale Architecture-combined with EMC® ScaleIO® and the open source CoprHD SDS controller-enables storage resource pooling and management that aligns with Intel’s Software Defined Infrastructure (SDI) vision for cloud environments, by matching the most performant and cost-effective resources to the mission- and business-critical nature of applications that are deployed and supported in a cloud or datacenter environment.

1.2 Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application programming interface</td>
</tr>
<tr>
<td>DAS</td>
<td>Direct attached storage</td>
</tr>
<tr>
<td>JBOD</td>
<td>Just a bunch of disks</td>
</tr>
<tr>
<td>MDM</td>
<td>Metadata manager</td>
</tr>
<tr>
<td>NIC</td>
<td>Network interface card</td>
</tr>
<tr>
<td>O&amp;M</td>
<td>Orchestration and management</td>
</tr>
<tr>
<td>PCIe*</td>
<td>Peripheral Component Interconnect Express*</td>
</tr>
<tr>
<td>SDC</td>
<td>ScaleIO® Data Client</td>
</tr>
<tr>
<td>SDI</td>
<td>Software Defined Infrastructure</td>
</tr>
<tr>
<td>SDS</td>
<td>Software-defined storage</td>
</tr>
<tr>
<td>SDSC</td>
<td>Software-defined storage controller</td>
</tr>
<tr>
<td>SLO</td>
<td>Service level objective</td>
</tr>
<tr>
<td>TCO</td>
<td>Total cost of ownership</td>
</tr>
<tr>
<td>VG</td>
<td>Volume group</td>
</tr>
</tbody>
</table>
2 Overview

Intel® Rack Scale Architecture implementations represent the next generation of software-defined infrastructure that can be dynamically provisioned from pooled compute, network, and storage resources. An Intel® Rack Scale Architecture-based solution will allow a customer to “right-size” end-to-end infrastructure needs according to anticipated workload performance and capacity requirements.

As cloud environments become the predominant IT infrastructure of choice (public, private, and hybrid), it is critical that new software-defined architectures have clearly defined business and technical goals regarding infrastructure flexibility, composability, and total cost of ownership (TCO). Intel® Rack Scale Architecture and Intel SDIs provide efficient, easy-to-use, standards-based RESTful management interfaces that technology partners and cloud service providers can integrate with their own tools and products.

The first fundamental capability that Intel® Rack Scale Architecture provides is the discovery and organization of compute, network, storage, and rack-level assets within and across racks in the datacenter: Resources like CPUs, hard disk drives (HDDs), network interface cards (NICs), memory, cooling systems, shared power distribution units, and more. Datacenter staff can organize these into logical resource pools based on relative similarities, such as performance, bandwidth, locality, etc. This organization and logical pooling of assets allows customers to select resources that match service-level requirements and policies, such as resiliency, performance, and capacity—without having to know a myriad of infrastructure details.

In summary, the Intel® Rack Scale Architecture provides the following:

- **User-defined performance:**
  - Choice of technology for performance and cost optimization.
  - Modularity and resource disaggregation to provide easier paths to upgrade.
  - Ability to match workload to optimal hardware and software.

- **Maximum utilization:**
  - Pooled resource and workload-driven assembly of specific systems.
  - Pooled storage for higher resource utilization.
  - Dynamic management of logical systems.

- **Interoperable solutions:**
  - Interoperable solutions and management architectures.
  - Standard protocols to manage multivendor solutions.
  - Interoperability testing and compliance.

Intel® Rack Scale Architecture logical resource pools can be used by SDS platforms, such as ScaleIO® (or open source alternatives). Higher level software-defined storage controllers (SDSC), such as the open source CoprHD controller, can provide the methods required to compose various disparate resources into resilient systems that can then be provisioned, matched, and managed according to detailed application-based SLOs and protocols.

§
3 The EMC® ScaleIO® Storage Personality

EMC® ScaleIO® is a scale-out SDS solution that leverages host-based internal storage to create a server area network (SAN) with performance comparable to or better than an external SAN, at a fraction of the cost and complexity. ScaleIO® installs lightweight software components on application hosts, which contribute attached disks and any other direct attached storage (DAS) resources to the ScaleIO® cluster by installing the SDS software. Hosts can then be presented volumes from the ScaleIO® cluster by leveraging the ScaleIO® Data Client (SDC). These components can run alongside other applications on any physical, virtual, or cloud server, and can leverage any type of storage media, such as disk drives, flash drives, PCIe* flash cards, or cloud storage.

Figure 1 EMC® ScaleIO® network diagram

ScaleIO® can be deployed as dedicated storage nodes only, or can greatly simplify the infrastructure by converging the storage, compute, and networking resources into a single, easily managed, scalable building block. The capacity and performance of all available resources can be aggregated and made available to every participating ScaleIO® server and application. Storage tiers can be created with media types and drive types that match the ideal performance or capacity characteristics to best suit the application needs.

Storage and compute resources can be added to or removed from the ScaleIO® cluster as needed, with no downtime and minimal impact to application performance. The self-healing, autabalancing capability of the ScaleIO® cluster ensures that data is automatically rebuilt and rebalanced across resources when components are added or removed, or if they fail. Because every server and local storage device in the cluster is used in parallel to process I/O operations and protect data, system performance scales linearly as additional servers and storage devices are added to the configuration.
Key ScaleIO® highlights:

- **ScaleIO® Data Client (SDC):** Installs on application server node and intercepts IO requests; block device driver.
- **ScaleIO® Data Server (SDS):** Contributes local storage to the cluster; and handles disk I/O, availability.
- **ScaleIO® Metadata Manager (MDM):** Manages monitoring and configuration; holds clusterwide component mapping, not in the data path.
- Deploys as two-layer (where application and storage/compute are in separate servers) or hyper-converged (where application, storage, and compute are in the same servers).
- Installs on physical (Windows, Red Hat, CentOS, SUSE), virtual (VMware, Hyper-V, KVM, XenServer), and cloud (OpenStack) environments.
- Provides two-copy mesh mirroring protection: Volumes, split into “chunks”, distributed on node cluster and mirrored across all disks for load balance and availability.
- Allows any node configuration to coexist and operate within the cluster.
- Involves all other nodes in the cluster in failed node rebuilds.
- Does not require capacity planning or migrations.
- Supports features such as writable snapshots, data obfuscation, QoS (IOPS, BW), thin provisioning, performance tiers, multitenant capabilities, autorebalance, flash cache, REST API, and more.

Intel® Rack Scale Architecture, through RESTful APIs, presents resource pools and assets that can be used by ScaleIO® to support various performance tiers, resiliency, and fault domains. ScaleIO® represents one of various storage personalities that can reside on the Intel® Rack Scale Architecture infrastructure, exemplifying and supporting the flexibility key value proposition premise.
4 CoprHD

CoprHD is the open source development project based on the EMC® ViPR® controller, which EMC will continue to offer as a commercial product. The CoprHD controller is an SDS controller whose primary function is to provide services that allow for the management and control of various storage resources that customers have deployed in their cloud infrastructure. By using a single “pane of glass” and logical control path, customers can manipulate and analyze the use of new and legacy storage platforms across object, file, and block protocols. In addition, CoprHD can interact with and service user-driven storage requests using a self-service portal, or can reside behind orchestration and management (O&M) tools that request storage resources using CoprHD APIs.

Figure 2 Management O&M tools
Key CoprHD highlights:

- Storage, manipulation, and analysis of unstructured data on arrays and commodity-based systems.
- Object: Works across various commodity-based systems-EMC and other file-based arrays; compatible with Atmos, Amazon S3, Swift APIs.
- HDFS: Builds a Big Data repository at scale and runs Hadoop analytic applications.
- Block: Automates provisioning; adds block pools to CoprHD Service Catalog; powered by ScaleIO®.
- Storage engine: writes object-related data (users, metadata and object location) to logical containers of contiguous disk space (128MB) called “chunks”
- Information is indexed in append-only pattern.
- Data protection on chunks that are snapped, journaled, and versioned; includes erasure coding; object recovery; HW monitoring; directs I/O to disks.
- Does not overwrite data, requires locking or invalidates cache.
- Write requests to same chunk happen simultaneously on different disk sets.
- Throughput takes advantage of all spindles and NICs.
- Small object payloads are aggregated in memory.
- Provides geo protection against site failures; asynchronous replication.

A critical aspect of CoprHD’s ability to adhere to application-based service requirements is timely awareness of the state of the physical infrastructure, including faults, performance challenges, and degradation, in addition to planned maintenance activities. Intel® Rack Scale Architecture management software provides this level of awareness, and through RESTful APIs makes this information available via push and pull queries.
5 Intel® Rack Scale Architecture

Intel® Rack Scale Architecture is a logical architecture that disaggregates compute, storage, and network resources, and introduces the ability to pool these resources for more efficient utilization of assets. It simplifies resource management and provides the ability to dynamically compose resources based on workload-specific demands. The new architecture brings significantly increased performance and lower total cost of ownership. More specifically, Intel® Rack Scale Architecture can:

- Accelerate service delivery through a software-composable system and rapid provisioning for applications.
- Improve operational efficiency through increased resource utilization and interoperability.
- Enable a smaller IT unit of required capacity, reduce SKU inventory, and enable resource-specific upgrades.

Figure 3 Basic Intel® Rack Scale Architecture benefits

Intel® Rack Scale Architecture uses compute, fabric, storage, and management modules that work together to build a wide range of virtual systems (Figure 4). The design uses four basic pillars, which can be configured based on the needs of your solution stack.

- **Pod Manager** for multirack management, includes firmware and software APIs that enable resource and policy management and expose the hardware below and the orchestration layer above via a standard interface.
- **Pooled systems** of compute, network, and storage resources, composed based on workload requirements.
- **Podwide storage** built on Ethernet-connected storage, uses storage algorithms to support a range of usages deployed as a multirack resource or storage hardware and compute nodes with local storage.
- **A configurable network fabric** consisting of hardware, interconnectivity with cables and backplane, and management software, supports a wide range of cost-effective network topologies, including current top-of-rack (TOR) switch designs and distributed switches in the platforms.
An important goal for Intel in 2015 is driving alignment to the Intel® Rack Scale Architecture management software framework that sits below orchestration layers, such as those provided by Microsoft®, VMware®, and open source solutions. The framework APIs will deliver critical capabilities like topology discovery, disaggregated resource management, and system composition support, using a single standards-based implementation across compute, network, and storage elements.

The Intel® Rack Scale Architecture development team will be working very closely with key partners to continue to create industry-standard specifications and extensions to the framework APIs. Intel's goal is to create a framework that exposes differentiated component technology, such as platform telemetry and control, in a consistent manner that will enable cloud service providers the ability to write applications that leverage these differentiated component technologies.
6 Intel® Rack Scale Architecture with EMC® ScaleIO® and CoprHD Demonstration

For demonstration and proof-of-concept purposes, Intel and EMC deployed ScaleIO® and the open source CoprHD controller on an Intel® Rack Scale Architecture Software Development Vehicle (SDV) known as Bulldog Creek. The Bulldog Creek SDV consisted of two compute trays, each containing four dual-socket Intel® Xeon® E5 family servers, with Haswell CPUs and DDR4 memory and an Intel high-bandwidth Ethernet switch (code named Red Rock Canyon). The Red Rock Canyon switches featured 40 GbE connections to each server node, and were connected to each other with a 100 GbE link and to a 40 GbE link to a top-of-rack (TOR) switch, as depicted in Figure 5.

Each Bulldog Creek SDV compute tray also contained a compute Pooled Systems Management Engine (PSME), accessible via a dedicated 1 GbE management backplane and connected to the tray components using hardware interfaces such as I²C. The PSME presents a RESTful interface which abstracts the underlying hardware and can be accessed by rack management entities such as the Intel® Rack Scale Architecture Pod Manager (PODM). This allows the PODM to inventory rack hardware resources and apply configuration and control power to assemble servers.

To provide storage, the rack contained two Quanta* JBR JBODs, with each JBOD containing two 14-bay backplanes stuffed with twelve 1 TB rotational hard drives and two 400 GB Intel® SSD P3700 solid state drives. Four of the servers in the rack were equipped with a SAS-capable Intel® RAID Controller RS3GC008, cabled via eight external ports to one JBOD backplane. The controller supported 12 Gbps operation, but the JBOD expanders functioned at 6 Gbps, so the SAS links from the servers to the JBODs ran at 6 Gbps.

Figure 5 Bulldog Creek SDV compute tray

After we assembled, connected, and powered up the hardware, the application software pooled storage resources and created an environment that supported Intel® Rack Scale Architecture's discovery of resources, composition of servers, and assembly into a cloud operating environment. We installed the CentOS Linux distribution (via a manual process using PXE boot) on each node equipped with storage hardware. We then installed the ScaleIO® SDS and MDMs using the ScaleIO® installation manager. (We may automate this process at a later date so that storage nodes can be quickly and easily deployed from the PODM or orchestration software.)

We loaded a fifth node with the Ubuntu* Linux operating system and the OpenStack controller to prove a cloud environment that servers can join once they are assembled.
We then set up a sixth node to run CoprHD in a virtual machine. Through CoprHD, we could manage pooled storage in the pod with the GUI, or the set of Intel® Rack Scale Architecture APIs accessible by PODM and cloud orchestrators.

For the purposes of this demo, ScaleIO® was deployed and linked to CoprHD. However, a real-world pod might contain many diverse storage software stacks, each with its own native management interfaces, all of which could be abstracted through CoprHD, to become available to users, the PODM, and orchestration layers. This makes CoprHD a very useful resource in an Intel® Rack Scale Architecture environment.

The CoprHD node may also be leveraged to host the Linux Logical Volume Manager (LVM), iSCSI Target Services, plus the Intel® Rack Scale Architecture storage PSME. The storage PSME provides a RESTful API and an abstraction layer, which allows the PODM to discover volume groups (VGs) and manage logical volumes (LVs) and iSCSI targets. This gives the PODM the ability to create boot devices for assembly, with diskless servers. Once block storage is attached to the storage PSME system (in this case a ScaleIO® volume, but it could also be block storage from other storage systems or even directly attached storage), it can be layered with a volume group and discovered by PODM. Logical volumes and iSCSI targets can be created on the block storage as needed, including copy-on-write snapshots of existing logical volumes, which help optimize performance and reduce capacity consumption. With this technique, diskless and stateless servers can boot directly from pooled storage into any operating environment desired, using PODM or orchestration layers (Figure 6).

Figure 6  Orchestration layers

At this stage we needed to intervene manually to stand up operating systems and software stacks. (In the future, this process may be automated to speed and ease deployment.)
Once the software was established, we had a path to complete pooling and abstraction of storage resources and software-based management via REST APIs at all layers of the storage stack (Figure 7):

- Future Intel® Rack Scale Architecture “deep discovery” processes will provide detailed information on storage hardware resources including controllers and drives, and where they are attached in the pod.
- Storage hardware can be pooled via some SDS like ScaleIO®, and storage pools can be abstracted via open source CoprHD. (This provides a single unified RESTful management interface available to PODM and orchestration.)
- For compute resources that are not yet booted, the Intel® Rack Scale Architecture storage PSME can be used to create bootable targets from pooled storage via RESTful APIs. The compute PSME can then be used to configure servers to boot from the targets and power them on, again entirely via RESTful APIs, so bare metal, stateless and diskless servers can be rapidly composed and assembled into functioning cloud nodes.
- Storage resources can be added directly from the storage pool by the Cloud Orchestrator through CoprHD for compute resources already booting into an operating environment.

For proof-of-concept and demonstration purposes, Intel implemented a simple asset discovery and server assembly GUI as a plugin to OpenStack Horizon. As shown in Figure 8, rack resources can be discovered and inventoried, providing a listing of servers with locations and configuration information such as CPU count and memory capacity.
Figure 8  **OpenStack rack resource inventory**

![OpenStack rack resource inventory](image)

Figure 9  **Sample JSON script sent with hardware description**

![Sample JSON script sent with hardware description](image)

Figure 9 depicts a hover-over showing the actual JSON sent to describe the hardware in a given location.
The node assembly page (Figure 10) shows how JSON is sent with hardware requirements and boot target information, allowing a node to be assembled from available hardware resources and powered up and booted from the pooled storage.

**Figure 10** OpenStack node assembly page

Note: The JSON implementation is still being refined; therefore, this example may not be entirely accurate compared to the final release.
6.1 EMC® ScaleIO®

The ScaleIO® GUI in Figure 11 shows a deployment of two pools, both spanning four servers and including a total of eight 400 GB Intel® SSD P3700 solid state drives and sixteen 1 TB rotational HDDs. The GUI allows users to monitor I/O workloads against the ScaleIO® pools and view overall cluster health and configuration information.

Figure 11 ScaleIO® user interface

Note: An alert is visible in this screenshot because this test deployment of ScaleIO® is not yet licensed. The cluster is otherwise healthy.
6.2 CoprHD

Using CoprHD as the SDSC in a datacenter requires a few configuration steps. For this demo, we used ScaleIO® as the block storage array that we managed and utilized (Figure 12). In summary:

- Discover different types of storage (file, block) from multiple providers.
- Create virtual arrays from these arrays.
- Pool virtual storage together (high performance, low performance, etc.).
- Automate management (creating volumes, exporting, etc.).
- Integrate with orchestration stacks, such as VMWare®, OpenStack, etc.

**Add physical assets that CoprHD will manage:**

1. (Storage providers and storage systems must be populated with the ScaleIO® information (IP address, password, etc.) so that CoprHD can discover the storage.)

**Figure 12 Add ScaleIO® to physical assets**

![Add ScaleIO® to physical assets](image-url)
Add a virtual array:

1. CoprHD abstracts the physical assets into a virtual array, which can be made up of one or more physical assets.
2. When you add the virtual array, the menu will have a button to Add a Storage System where you will select one or more physical storage systems to be include in the virtual array.

Add a virtual block pool:

1. Similar to abstracting the physical storage systems into virtual arrays, you must create virtual pools of storage which can be provided by multiple virtual arrays. These pools should have similar characteristics, such as performance or hardware.
2. For a single ScaleIO® installation, there will only be one virtual block pool.
Create a project:

1. Once the physical assets have been virtualized into a virtual array and virtual pools (Figure 14), a project should be created, which will have storage quotas and access lists for users (Figure 15).

Figure 14   Add a virtual array

![Virtual array interface](image1)

Figure 15   Add a project

![Project settings interface](image2)
Create volume from service catalog:

1. Storage can now be allocated to a project from a virtual array/pool and provided to a user (Figure 16).

Figure 16 Service catalog

2. Select block storage services (Error! Not a valid bookmark self-reference.).
3. Select create block volume (Figure 18).

4. Fill out the volume options to include the virtual array, virtual pool, project, name, and size desired (Figure 19). Consistency group and number of volumes can be left at their default values for now.
Figure 19  Block volume creation options
Once the block volume has been created, it will be visible in the Resources menu (Figure 20).

Although this was a simple walkthrough of how a single volume can be created using CoprHD communicating with ScaleIO®, it should be evident that a storage admin tasked with managing storage arrays from multiple different vendors would truly benefit from CoprHD and its ability to create virtual arrays and pools from a variety of storage systems using a single interface.
7 Future Directions

Intel® Rack Scale Architecture pooled storage provides an automated method for customers to organize their resources according to the physical attributes of the individual nodes. As additional telemetry and node-based predictive analytics capabilities are added, closer adherence to application SLOs will be possible.

Today's architectures are roughly based upon the knowledge acquired by individual operators that have deployed and observed how their infrastructure deployment does and does not support performance, cost, and capacity requirements for a given application. However, as the migration between private, hybrid, and public clouds becomes a standard mode of operation, knowledge about the behavior of individual applications on dedicated infrastructure will quickly become obsolete. In addition, getting the best efficiency out of a cloud infrastructure implementation is imperative for large scale cloud deployments, as idle time translates into cost overhead and loss of profitability.

Using policy-based service-level requirements as starting points, solutions such as the combination of Intel® Rack Scale Architecture, EMC® ScaleIO®, and CoprHD controller will be able to:

- Utilize infrastructure telemetry provided by Intel components.
- Associate the information with real-time and predicted performance.
- Intelligently redeploy applications and associated data to evolve from today's fault management paradigm to a fault-prevention mode of operations.