IBM Spectrum Protect

In-the-Cloud Deployment Guidelines with Amazon Web Services

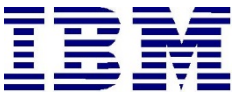
Document version 1.2

James Damgar

Daniel Benton

Jason Basler

IBM Spectrum Protect Performance Evaluation



CONTENTS

LIST OF FIGURES

LIST OF TABLES

Introduction

1.1 Purpose of this Paper

This document introduces possibilities for integrating an IBM Spectrum Protect server with an **Amazon Web Services (AWS)** cloud computing system. You can use the configurations as starting points to deploy a large, medium, or small system (as defined in the IBM Spectrum Protect Blueprints) or an extra-small system. With the goal of achieving a target daily ingestion rate (corresponding to a large, medium, small, or extra-small deployment), configuration possibilities are offered so that you can get a sense of the relative CPU, memory, disk, and network capabilities that are needed to satisfy requirements. In addition, a survey of options for fulfilling these needs is provided. Certain cloud instance and disk types offered by providers might be sufficient in some areas while lacking in others. You must recognize where system bottlenecks might arise that could limit IBM Spectrum Protect capability.

Use this paper as a **starting point for guidance** about where to deploy an instance of the IBM Spectrum Protect server within an Amazon Elastic Compute Cloud (Amazon EC2) dedicated or shared compute instance with the goal of eventually storing the bulk of primary backup and archive data on cost-effective object storage. In the case of Amazon, this means Amazon S3 (Simple Cloud Storage Service). This goal can be accomplished by configuring an IBM Spectrum Protect cloud-container storage pool with (block disk-based) accelerator cache. Two approaches are generally available for storing data in an IBM Spectrum Protect cloud-container storage pool: a direct-to-cloud approach or a disk-to-cloud tiering approach.

For **direct-to-cloud** architectures, backup data is ingested directly into a **cloud-container storage pool** with a performant accelerator cache disk location tuned for a system's ingestion workload as the initial "landing spot" (for more information, see Sizing the Cloud Accelerator Cache). Data is then immediately, asynchronously transferred to object storage while further data is also ingested into the disk staging area (also known as *overlapped I/O*). The key consideration here is to determine the performance characteristics that the disk staging area must provide to allow for this mixed write-and-read behavior to ensure that ingestion targets are met. A *Cloud Cache and Object Storage Benchmarking* guide and "Cloud benchmarking tools" packages are provided along with this paper to assist in benchmarking both the cloud accelerator cache and object storage system from a prospective host server.

In contrast, **disk-to-cloud** tiering architectures make use of IBM Spectrum Protect **storage rules** to demote data from one or more (usually small) directory-container storage pools (a disk tier) to a cloud-container storage pool. Backup data is initially ingested into a directory-container storage pool and later a portion of this data is moved asynchronously to a cloud-container storage pool. This can be done with either age-based tiering (available as of IBM Spectrum Protect Version 8.1.3) or tiering by backup state (available as of IBM Spectrum Protect V8.1.6). With IBM Spectrum Protect V8.1.6, a combination of storage rules and storage subrules can be used to facilitate more granular tiering behavior between the disk and object storage tiers, allowing for flexibility and filtering by node and node file space, for example. Some guidance is given in this paper to assist with determining whether tiering is suitable for your workload demands and characteristics. (For an introduction to the tiering features that are available in IBM Spectrum Protect, see Tiering data to cloud or tape storage in the online product documentation.)

With AWS, solutions involving disk-to-cloud tiering can be cost prohibitive due to the higher cost of Amazon Elastic Block Store (EBS) disk, which might have to be provisioned to support the disk tier, in contrast to the relatively lower cost of Amazon Simple Cloud Storage Service (S3) object storage. However, restore requirements of a solution may be such that having operational recovery data on a fast-performing disk tier is worth this additional cost. Further guidance is provided in the following section regarding considerations for using a direct-to-cloud or tiering approach. The architectures referenced within this paper use a direct-to-cloud approach but can be adjusted to conform to the requirements of a tiering architecture.

1.2 Considerations for Disk-to-Cloud Tiering Versus Direct-to-Cloud Data Movement

The primary advantage of the **tiering** model is that operational recovery data can be preserved on a localized, fast disk tier for rapid recovery while older copies of data or data intended for long-term retention can be demoted to object storage, which is typically more affordable. The tiering model can also be used as an alternative to the direct-to-cloud model with a relatively small disk tier footprint (not strictly for operational recovery purposes). When the `TIERDELAY` parameter is set to 0, age-based tiering can be used to tier each day's worth of ingested client data after it is ingested (after the backup operation is completed). In this case, potentially less expensive disk can be provisioned for use by the small disk container pool tier because no ingest and cloud transfer input/output (I/O) operations occur in parallel. Tiering can be run serially after the completion of data ingestion during scheduled windows with less or no contention for this disk; the disk area can be cleared in preparation for the next day's ingestion.

The same ingestion targets can be satisfied with the disk-to-cloud tiering model as with the direct-to-cloud model, assuming that the direct-to-cloud approach makes use of an accelerator cache and overlapped data ingestion.

Restriction: To implement cloud tiering, you must provision enough disk space to hold a full day's worth of ingested data (plus some buffer) to avoid failed backup operations. The same underlying disk technology can be used in both cases. If, however, you plan to use disk-to-cloud tiering to hold one or more days' worth of operational recovery data within a

container pool disk tier, the instance disk capacity might have to be much greater, with the caveat that a slower-performing disk might be sufficient for this case. In all cases, you must understand the ingestion targets (after data deduplication and compression) to determine a daily disk capacity for a transient disk case. Meanwhile, operational recovery requirements in terms of the number of days' worth of recovery data (after deduplication and compression) should be determined to further size a disk container pool with tiering to cloud if necessary.

With the **direct-to-cloud** model, you can minimize local block storage capacity. This is an advantage because local block storage can be cost prohibitive in cloud-hosted environments.

1.2.1 Cloud Accelerator Cache Considerations

Beginning with IBM Spectrum Protect V8.1.2, data ingestion from clients is throttled if the accelerator cache area is near capacity. This feature makes it possible for this disk cache location to be **underprovisioned** from a capacity standpoint in that the disk cache location does not have to be sized large enough to hold a full day's worth of deduplicated and compressed data. However, the accelerator disk still must be performant enough in terms of input/output operations per second (IOPS) so that client data ingestion and replication target activity can be completed in a timely manner. In the end, you have to compare costs to determine whether larger capacity, less-expensive disk with tiering has an advantage over a direct-to-cloud cache model for a given environment, ingestion rate, and recovery objective.

Restriction: If you plan to use the direct-to-cloud ingestion model, the cloud accelerator cache should be sized large enough to hold at least two times the largest front-end object being ingested. For example, if a 512 GB object is to be ingested directly into a cloud-container storage pool, the cloud accelerator cache should be at least 1 TB in size. Similarly, if 5 client sessions will be backing up 100 GB files each at the same time, the cloud accelerator cache should be sized to at least 1000 GB (5 clients x 100 GB files x 2). This is because the IBM Spectrum Protect server will attempt to "reserve" space in the cloud accelerator cache for in-flight ingestion until ingestion is completed for those objects and their database transactions are committed to the server. By default, this processing assumes no deduplication or compression savings and attempts to reserve the total front-end amount of data to ensure sufficient storage capacity.

Beginning with IBM Spectrum Protect V8.1.6, a server option can be used to influence this behavior. The `PreallocReductionRate` server option can be used to give the server a "hint" about the expected reduction ratio for ingested data and cause the server to reserve less physical space in the container storage pool. For example, setting this option to 5 will cause the server to assume a 5:1 data reduction rate for front-end to back-end data so that only 1 unit of back-end space will be reserved for 5 units of front-end protected data. This option can range from 1 (the default, no reduction) to 8 (an 8:1 assumed reduction). Use this option only when a smaller cloud accelerator cache is desired and data reduction rates are certain. If the storage pool has inadequate space, backup failures can occur.

1.2.2 Workload Limitations and Considerations with Tiering

Not all client workloads are suitable for a disk-to-cloud tiering model. Tiering by age (as of IBM Spectrum Protect V8.1.3) allows for the demotion of *backup* objects that have reached the specified age threshold. Inactive backup generations that are older than the specified age are transitioned to object storage. Tiering by state (as of IBM Spectrum Protect V8.1.6) allows for the demotion of backup objects that have reached the specified age threshold and are *inactive* within the server. Active backup objects are preserved on the disk tier, while inactive copies of backup objects are transitioned to object storage.

Disk-to-cloud tiering is **suitable** for client workloads that have **low** data deduplication rates (backup generations differ greatly). In this case, data is highly unique between backup operations. When a backup generation is tiered to object storage, the deduplicated extents (chunks) that make up that object have their references decremented on the source directory-container storage pool. In this case, reference counts are likely to be low and more deduplicated extents are likely to be removed from the disk tier as objects are tiered and space is released.

Disk-to-cloud tiering **might not be suitable** for client workloads that have a **high** data deduplication rate. In this case, data is not very unique between backup generations and many shared deduplicated extents are referenced by multiple object generations. Even though an object can be tiered by a storage tiering rule, because the object shares many extents with other objects (which might still be active), a large proportion of the object's data will not be removed from the disk tier (although it will be copied to the object storage tier).

The following figures illustrate how data movement with disk-to-cloud tiering can occur. Figure 1 depicts a scenario in which multiple versions of three backup objects (A, B, and C) have been ingested and are stored in a directory-container storage pool on disk. Dotted lines represent references to deduplicated extents (colored, numbered boxes). With the tier-by-state option, the inactive object copies (shown in the gray rectangle) would be tiered to a cloud-container storage pool.

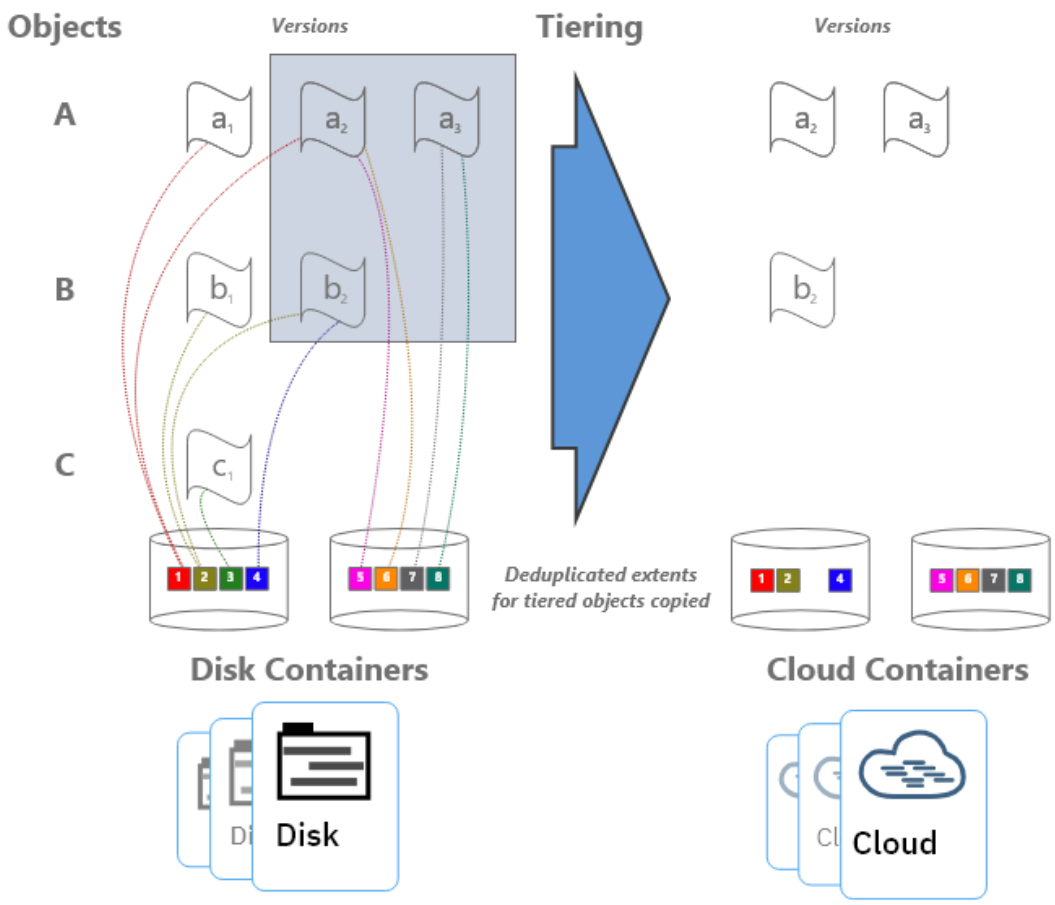


Figure 1: Disk-to-cloud tiering, before and after

Figure 2 depicts the situation after tiering is completed and the REUSEDELAY parameter value of the source directory-container storage pool is exceeded (so that deduplicated extent removal for extents with zero reference count can occur).

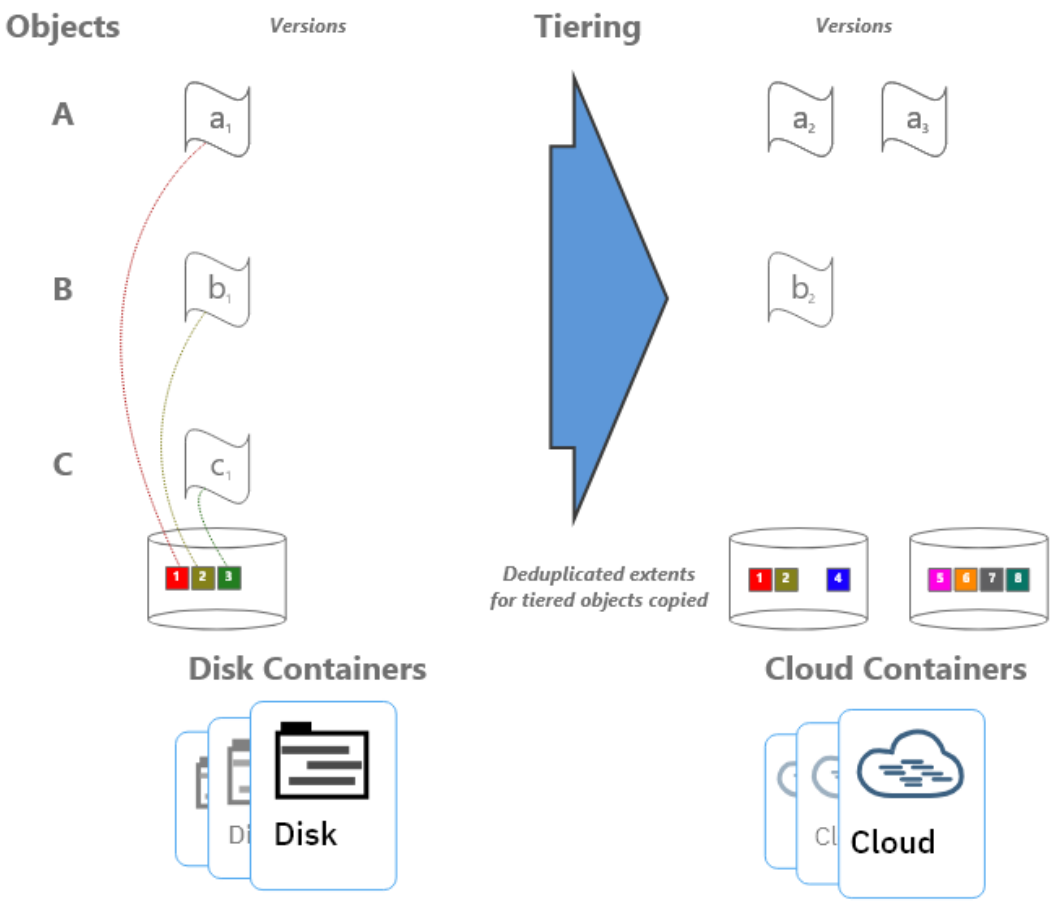


Figure 2: Disk-to-cloud tiering, after tiering

Notice that deduplicated extents 1 and 2 remain on disk even after tiering and extent cleanup have occurred. This is due to the fact that those extents are shared between the active and inactive backup copies. If many deduplicated extents are shared by objects (a high duplicate data rate with high data deduplication ratios), it is more likely that data will remain on disk, even after backup objects have been tiered at an IBM Spectrum Protect inventory level. Keep this factor in mind when you consider a disk-to-cloud tiering model and when you size an environment.

For workloads that deduplicate well from day to day, there will be many shared extents across backup and archive generations and a smaller capacity footprint on tiered object storage as a result because these backup and archive generations will also share many extents in the cloud-container storage pool. For workloads that deduplicate poorly day to day (highly unique data change each day), there will be few shared extents across backup and archive generations and potentially a larger capacity footprint on tiered object storage because these backup and archive generations will each point to (more) unique data in the cloud-container storage pool.

If the primary motivation for using disk-to-cloud tiering is rapid recovery of operational data, a tiering model might provide the best approach. You must understand the nature of the client workload to accurately size the directory-container storage pool on disk.

1.2.3 Amazon S3 Intelligent-Tiering

Beginning with IBM Spectrum Protect V8.1.8, you can enable Amazon S3 Intelligent-Tiering for objects in IBM Spectrum Protect cloud-container storage pools. When you issue the `DEFINE STGPOOL` command to define a cloud-container storage pool with Amazon S3, you can specify a new parameter, `CLOUDSTORAGECLASS` (and update the parameter value later by using the `UPDATE STGPOOL` command). Setting this parameter to a value of `AUTOMATICVENDORTIERING` will cause all new cloud-container storage pool objects that are uploaded to S3 to be sent to the Intelligent-Tiering storage class (as opposed to the default Standard storage class). Objects with the `AUTOMATICVENDORTIERING` storage class type can be transitioned automatically to the Amazon S3 Infrequent Access tier if the object has not been accessed for 30 consecutive days. This functionality offers storage capacity savings for Amazon S3 users. IBM Spectrum Protect data that is already deduplicated and compressed can be further transitioned to a lower-cost storage tier within S3, provided that restore or other read requests that would involve deduplicated extents within the transitioned object are infrequent. With Amazon S3 Intelligent-Tiering, objects that are read are automatically moved back to the frequent access tier. For more information about Amazon Intelligent-Tiering, see References [10].

1.3 Cloud Deployment Patterns

The described configurations can be used as starting points in situations where the IBM Spectrum Protect cloud instance will be a **primary server and in situations where it is used as a replication target**. In scenarios where the cloud-based instance is a replication target, adequate “public” network capability might be necessary to satisfy replication throughput requirements. AWS Direct Connect can be used to establish a dedicated 1 Gbps or 10 Gbps network connection from an on-premises data center to AWS to facilitate efficient IBM Spectrum Protect replication or backup processing from peer servers or clients outside the AWS infrastructure.

Generally, IBM Spectrum Protect deployments making use of cloud-based object storage will align with one of the following three patterns:

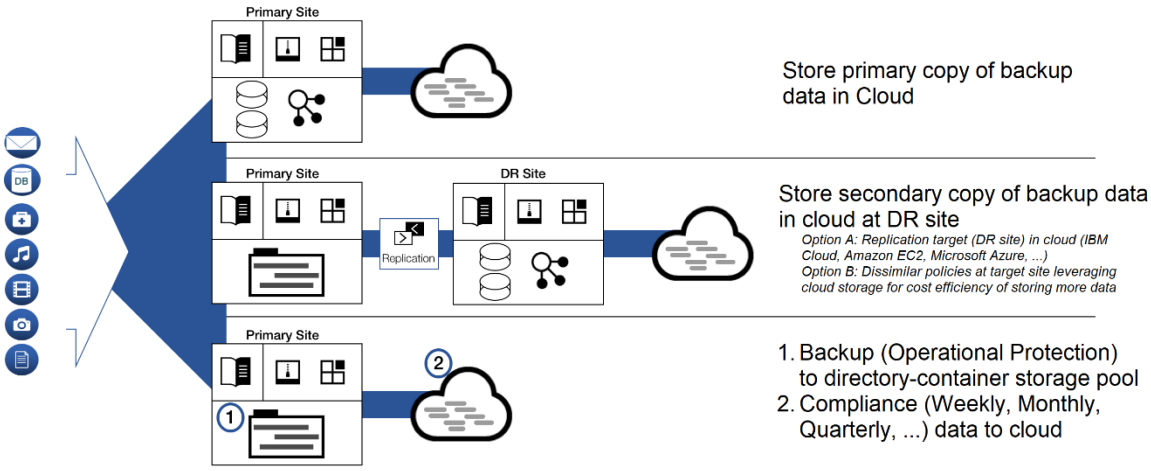


Figure 3: Deployment patterns

In the figure, the first deployment pattern could involve an IBM Spectrum Protect server that is installed on premises or in Amazon on an EC2 instance, with primary backup and archive data landing in object storage immediately. The positioning of the IBM Spectrum Protect server in relationship to clients could be one critical decision point when you consider whether to have a server instance on premises or within AWS. This pattern could involve use of a direct-to-cloud architecture with accelerator cache or a small disk container pool with immediate tiering to a second cloud-container storage pool without accelerator cache.

The second deployment pattern would make use of cloud-based Amazon S3 object storage at the secondary disaster recovery (DR) site. This DR server could be installed at an on-premises site or on an Amazon EC2 instance. In the latter case, sufficient wide area network (WAN) bandwidth between the primary and secondary sites is required for acceptable performance. Much like the first deployment pattern, here the IBM Spectrum Protect server at the DR site could make use of a direct-to-cloud topology with a cloud-container storage pool featuring accelerator cache, or it could use a small disk container pool landing spot with immediate tiering to a cloud-container storage pool backed by object storage.

The third deployment pattern features specific use of disk-to-cloud tiering, available with IBM Spectrum Protect V8.1.3 and later, to allow for operational recovery data to reside on faster performing disk storage. Data that is older, archived, or both would be tiered to cloud-based Amazon S3 object storage after a specified number of days. This deployment could also be implemented at an on-premises site or within an Amazon EC2 instance. However, the additional cost of having a larger capacity disk container pool should be factored into cost estimates with an in-the-cloud solution.

A **combination** of approaches is also possible within the same deployment. For example, a cloud-container storage pool could be configured with accelerator cache disk and made to store long-term retention or compliance archives. A directory-container storage pool could be configured as a disk tier for normal backups, and a tiering relationship could be set up so that operational recovery data (for example, backups from the previous 7 days) is kept on this disk tier, while older data is demoted to the same cloud-container storage pool.

The same cloud-container storage pool can be a direct backup target and a tiering target. However, if the pool is a direct target of a backup-archive client, the pool must be configured with accelerator cache disk.

1.4 Cloud Environment Considerations

For Amazon EC2, you can choose one of multiple potentially suitable instance types to satisfy an IBM Spectrum Protect server requirement. You do not have complete flexibility in terms of the type and proportion of CPU and memory resources available for an instance. However, standard instance types are available, which can roughly match recommended server requirements. Take care to select instances that have the required **CPU and memory** resources to support inline server data deduplication, compression, encryption, and cloud application programming interface (API) operations. The instances also must have the associated storage area network (SAN) disk (Amazon EBS) and network (Ethernet) capability to support the desired workload.

With Amazon Elastic Cloud Compute (EC2) instances, you can alter the instance type later, when resource requirements might be clearer. The process is fairly straightforward for Elastic Block Store (EBS) backed instances using Amazon's web portal. However, there are some caveats to keep in mind. For details, see the Amazon documentation under References [3]. In particular, experience shows that driver issues can occur when attempting to resize Windows-based EC2 instances after initial provisioning. To help avoid complications, use Red Hat Enterprise Linux or Ubuntu Linux instances in Amazon EC2 with IBM Spectrum Protect.

1.4.1 Importance of Adequate Sizing

Ingested backup data reaching cloud accelerator cache or the initial disk container pool tier requires the use of block storage allocated to the cloud server. IBM Spectrum Protect database activity also uses some level of throughput and elevated I/O operations during workload processing. Therefore, disk **I/O capability** and instance-to-disk throughput considerations must be evaluated when choosing and configuring an in-the-cloud IBM Spectrum Protect instance. With a final ingestion point on object storage via a cloud-container storage pool, the **Ethernet capability** of the instance and the nature of the network between the instance and the Amazon S3 object storage endpoint must be kept in mind. As part of this, consider how the front-end client data workload deduplicates and compresses data into the back-end stored quantity.

Certain Amazon instances might or might not have access to dedicated bandwidth to attached EBS disks. A lack of access can create a bottleneck in the IBM Spectrum Protect server's database operations. Certain instances might have limited throughput over Ethernet, and this limitation could hamper ingestion and restore throughput with object storage. During the planning phase, consider how ingested data will be **reduced by data deduplication and compression** when the data moves to the back-end storage location. These factors will help you estimate how much back-end data must be moved within a certain time window (measured in hours) and can help predict the throughput (megabytes per second or terabytes per hour) that the Ethernet network and object storage endpoint require to satisfy ingestion requirements. Generally, **10 Gbps** Ethernet capability to private

Amazon S3 storage endpoints is required for large, medium, or small Blueprint ingestion targets, while 1 Gbps is sufficient for extra-small targets.

Beginning with IBM Spectrum Protect V8.1.3, the server automatically throttles client backup operations if the cloud accelerator cache portion of a cloud-container storage pool is nearing full capacity. As a result, it is not mandatory to configure cloud accelerator disk cache space that would be large enough to hold a full day's worth of backups (after data deduplication and compression). However, disk benchmarks should be run to ensure that the anticipated back-end workload that an IBM Spectrum Protect server is expected to support will not result in this disk location being the primary bottleneck of the system (see Disk Benchmarking). In practice, any planned deployment should be validated to ensure that it will meet performance requirements.

1.4.2 Linux Logical Volume Manager (LVM)

The described reference architectures use either the Red Hat Enterprise Linux or Ubuntu Linux operating system. If you want to deploy a solution on Linux based operating systems, the preferred method is to use the **Linux Logical Volume Manager (LVM)** for the cloud accelerator cache disk and, optionally, the IBM Spectrum Protect database archive log disk (when more than one physical disk is utilized). The remaining IBM Spectrum Protect disk components can be satisfied with file systems formatted on directly mounted Amazon EBS block disks. The overlapped I/O pattern experienced with concurrent backup ingestion activity and transfer of data to object storage can lead to "hot spots" on disk when more than one storage pool directory is defined for a cloud-container storage pool as accelerator cache. To help avoid a throughput bottleneck, you can configure a single logical volume to span all physical volumes assigned for use as cloud accelerator cache. Furthermore, the preferred method is to use a stripe size of **16 KiBytes** for the single logical volume and ensure that the number of stripes matches the number of physical disks. For more information, see Disk Setup Commands for Linux Deployments.

The use of Linux LVM to logically stripe across several Amazon EBS disks should not be depended upon to increase the durability of the underlying storage, as Amazon EBS disks are already redundant within an Amazon Availability Zone and so do not benefit from the recovery characteristics of LVM striping or RAID.

1.5 References to Physical IBM Spectrum Protect Blueprints

Throughout this paper, the physical server specifications contained in the latest (V4.1) ***IBM Spectrum Protect Blueprint and Server Automated Configuration for Linux x86*** document (also known as the IBM Spectrum Protect Blueprints) are referenced as targets for CPU and memory configurations matching small, medium, and large server builds. For more information about the Blueprints, see References [1]. The intention with the server builds outlined here is to provide systems capable enough from a CPU, memory, disk, and Ethernet point of view to approach Blueprint-level ingestion capability. Although different instance types can be used to satisfy the same requirements, and deviations from the Blueprint specifications mentioned in this document are possible, the disk specifications should be followed to ensure appropriate ingest and database scalability and performance.

As a reference, the following table indicates the throughput, capacity, CPU, and memory targets for each of the referenced Blueprints. The values for total managed data and daily

ingest data are for the block storage Blueprints. These ingestion targets assume an 8-hour backup window.

Table 1: IBM Spectrum Protect physical Blueprint targets (V4.1, Linux x86)

Sizing category	CPU	RAM memory	Total managed data (front end)	Daily ingest data (front end)
Small	16 cores	64 GB	60 TB – 240 TB	Up to 10 TB per day
Medium	20 cores	128 GB	360 TB – 1440 TB	10 – 30 TB per day
Large	44 cores	384 GB	1000 TB – 4000 TB	20 – 100 TB per day

Although not defined explicitly in the physical Blueprints, the extra-small cloud Blueprint systems target up to 10 TB or more of total managed (front-end) data with a daily ingestion rate of up to 1 TB, or more, per day.

1.6 Database Backup Capacity

The configurations in this document are provisioned with enough IBM Spectrum Protect **database backup disk** space to hold two days' worth of full database backups in the worst case (an IBM Db2 database consuming close to its resident capacity). This approach was taken to avoid the excessive attached disk cost associated with provisioning excess cloud disk space for storing many days' worth of backups. For example, with Amazon EBS, GP2 type volumes cost $0.10 per GiByte-month in the US West (Oregon) Region based on provisioned space. Compare this with the $0.023 per GiByte-month in the same region for S3 object storage where pricing is based on actual storage used. Not only is unused provisioned disk space a deterrent to cost savings, the actual rate charged for this space is much more than object storage considering that the data involved (database backups) is archive-like in nature.

The IBM Spectrum Protect server is presently incapable of storing database backups directly to object storage. For those desiring recovery protection for a longer period (for example, 7 – 10 days), the database backup disk space outlined here might have to be increased. Alternatively, a strategy can be employed outside of IBM Spectrum Protect whereby the disks that underlie the database backup disk can be snapshotted at the cloud service provider's level and stored to object storage after a database backup is completed. This may be driven manually or, more likely, in a scripted fashion by using cloud provider APIs from within the instance itself. This has been done with success in IBM GTS with Amazon based IBM Spectrum Protect deployments.

If you use this method, the disk space within the database backup disk location could be cleared in preparation for the next day's database backup, allowing for only two days' worth of footprint on provisioned cloud block disk. The reason that two days' worth of capacity is required and not one is that it is impossible to delete the "only" database backup copy until after a new one is created. These object storage-based disk snapshots would need to be handled with great care and security. If you want to recover a database

to a point N days in the past, the disk snapshots for the database disk volumes would have to be restored from object storage before normal IBM Spectrum Protect database backup recovery could commence.

1.7 Server Maintenance Scheduling Considerations

The *IBM Spectrum Protect Blueprint and Server Automated Configuration for Linux x86 V4.1* document provides a detailed breakdown of the procedure for setting up IBM Spectrum Protect server maintenance schedules (see References [1], Chapter 5). Use this information as a reference for establishing a maintenance schedule on cloud-hosted servers.

For an IBM Spectrum Protect server in AWS that is serving as a replication target, a replication window and schedule might have to be established. For servers using the direct-to-cloud model, where primary backup data is ingested directly into a cloud-container storage pool, a replication window might not be required if this server is not a replication target server because a cloud-container storage pool cannot be used as a replication source. In this case, redundancy requirements for the ingested client data can be met by the inherit redundancy of Amazon S3 object storage.

For an IBM Spectrum Protect server in AWS that is using the disk-to-cloud tiering model, a replication source strategy might be required. Replication can help to protect client data objects in the disk directory-container storage pool that have not yet been tiered (demoted) to object storage because only one copy of that data is present. To prevent excess data from being stored (pinned) to the disk tier, verify the following items:

- The source replication server (used for disk-to-cloud tiering) should be configured with a longer retention policy than the target replication server. In other words, data should be retained for a longer period on the source replication server.
- The retention policy that affects client node data on the target replication server should match the value of the `TIERDELAY` parameter of the storage rule responsible for tiering the same client node data on the source server.

In general, the server that is used for disk-to-cloud tiering (whether it be the source replication server or the target replication server) should be the server with the longer retention policy for the client nodes that are affected by the tiering storage rule.

1.8 Session Scalability by Blueprint Size

The **IBM Spectrum Protect Blueprint and Server Automated Configuration for Linux x86** document describes how to set the IBM Spectrum Protect server option `MAXSESSIONS`, based on Blueprint system size:

- Small system: 250 maximum simultaneous client sessions
- Medium system: 500 maximum simultaneous client sessions
- Large system: 1000 maximum simultaneous client sessions

(For more information about the Blueprint configurations, see References [1].)

The actual throughput scalability of a cloud-based solution depends on many factors, including the configured disk capability and capacity of the system, the amount of CPU and memory resources available on the system, and the relative rate of data deduplication and compression for the dataset that is ingested into the server. Larger objects, which feature a larger deduplicated extent size (for example, 250 - 350 KiBytes, or more) and which do not deduplicate or compress well (for example, less than 10%), will result in less database and computation (CPU) overhead, but will utilize more disk and network bandwidth. The logical reduction of front-end client data to the physical back-end data (which is actually written out and stored to disk and object storage) means that the disk, network, and object storage components will be stressed to a higher degree as client/server session counts increase. Memory usage by the IBM Spectrum Protect server might also be greater. As session counts increase, these components are likely to become a system bottleneck, limiting front-end throughput.

Objects that feature smaller, deduplicated extent sizes (for example, 60 - 100 KiBytes or similar) and that deduplicate and compress well (for example, 50% data deduplication with 50% compressibility) will result in less network, disk, and object storage bandwidth used, but will lead to more database and computation overhead to facilitate these data reduction operations. As session counts increase, CPU and database-related memory are likely to first become limiting factors for these data types. In general, the more successfully data can be deduplicated and compressed (and therefore the greater the data reduction from front-end to back-end data), the greater the number of feasible client sessions. The following table indicates a reasonable range of client session counts based on system size and data type, as well as the likely limiting factor for the system as the high end of the range is approached. For more information about these data types, see Throughput Measurements and Results.

Table 2: Preferred ranges of maximum values for client session counts

Cloud system size	**Large object, poor data deduplication and compression[1]**	**Large object, good data deduplication and compression[2]**	**Large object, small extent size, good data deduplication and compression[3]**	**Small object, poor data deduplication and compression[4]**
Extra small	10 – 50	25 – 50	10 – 50	10 – 50
Small	50 - 100	100 - 200	50 - 100	50 - 100
Medium	100 - 200	200 - 400	100 - 150	100 - 150
Large	300 - 400	400 - 500	150 - 200	150 - 200

Limiting factor at scale	Network, disk, object storage bandwidth, memory	CPU, memory or network, disk, object storage bandwidth	CPU, memory	CPU, memory

[1] This model uses 128 MiByte objects, 250 - 350 KiByte extents, and <10% data deduplication and compressibility. Full backup operations are used with pseudo random data or data that cannot be easily deduplicated or compressed. For example, this model can be applied to encrypted data.

[2] This model uses 128 MiByte objects, 150 - 200 KiByte extents, and 50% data deduplication and compressibility. For example, this model can be applied to virtual machine backups.

[3] This model uses 1 GiByte objects, 60 - 100 KiByte extents, and 50% data deduplication and compressibility. For example, this model can be applied to database image backups.

[4] This model uses 128 KiByte objects and <10% data deduplication and compressibility. For example, this model can be applied to file server data and other small files or objects.

Often, a diminishing rate of return in regard to throughput is experienced when 50 - 100 total client sessions are exceeded, regardless of data type. More sessions are possible and might be warranted, given the client schedule or other requirements. However, aggregate gains in total throughput of a single IBM Spectrum Protect instance might not be substantial past this point.

Amazon EC2 Compute Configurations

For an overview of this Amazon web service, see Amazon EC2.

The following configurations represent preferred IBM Spectrum Protect deployments within the Amazon Web Services infrastructure. These deployments make use of **Amazon Virtual Private Clouds (VPCs)** within a desired Amazon region with a VPC endpoint configured for the Amazon S3 service. The Amazon S3 bucket utilized by each instance for IBM Spectrum Protect usage is created within the same Amazon region and is accessed by the IBM Spectrum Protect server via HTTPS. Amazon guidelines call for creating a **VPC endpoint for the S3 service** so that instances accessing the public endpoint address (for example, using an IBM Spectrum Protect cloud-container storage pool with a cloud URL of https://s3.us-west-2.amazonaws.com for US West Oregon) will have traffic routed through Amazon's internal network rather than being exposed to a broader WAN. This approach benefits both the reliability and latency of IBM Spectrum Protect cloud operations.

For circumstances in which the IBM Spectrum Protect server is provisioned within an Amazon EC2 compute instance and this server is the target server for storage pool protection and node replication operations from a source server located at an on-premises data center, additional network bandwidth might be needed to satisfy replication requirements. Consider using AWS Direct Connect to establish a dedicated, high bandwidth link from your premises to the AWS-based IBM Spectrum Protect instance running in EC2. Consider this option if your anticipated back-end replication workload will cause the needed throughput to exceed the bandwidth capacity between the replication pair. Amazon Direct Connect can be used to establish a dedicated 1 Gbps or 10 Gbps

network connection from an on-premises data center to AWS in order to access Amazon Virtual Private Cloud (VPC) as well as public AWS services, such as Amazon S3.

For example, if an IBM Spectrum Protect source server were to ingest 40 TB of front-end data that reduced to 10 TB after (2:1) data deduplication and (2:1) data compression, that 10 TB of back-end data would have to be replicated during the daily storage pool protection and node replication window. A 1 Gbps dedicated link can support approximately 3.5 TB per hour of throughput. Amazon Direct Connect offers both 1 Gbps and 10 Gbps options with the additional benefit of being charged an AWS Direct Connect data transfer rate that is below the rate charged for incoming internet data transfer. See References [6].For each of the instances, isolated throughput measurements should be conducted with an S3 API tool to determine maximum throughput to collocated S3 bucket storage within that region.

Table 3: AWS, large configuration

Cloud component	**AWS component**	**Detailed description**	**Quantity**
Server and network	m5.16xlarge or m5a.16xlarge Amazon EC2 instance (dedicated)	64-core Intel Platinum 8175 @ 3.10GHz or AMD EPYC 7000 @ 2.5GHz	1
		256 GB RAM	
		EBS disk optimized with 10000 Mbps EBS throughput	
		20 Gbit Ethernet connectivity	
	Virtual Private Cloud (VPC)	Instance placed in VPC; endpoint created for Amazon S3 service in the same AWS region	1
	Operating system	Red Hat Enterprise Linux or Ubuntu Linux server	1
Block storage	100 GB EBS GP2 volume	Operating system disk	1

	100 GB EBS GP2	IBM Spectrum Protect instance disk	1
	2000 GB EBS GP2 volume	IBM Spectrum Protect database disk	8
	16384 GB EBS ST1 volume	IBM Spectrum Protect database backup disk	2
	665 GB EBS GP2 volume	IBM Spectrum Protect database active log disk	1
	4096 GB EBS ST1 volume	IBM Spectrum Protect database archive log disk	2
	3300 GB EBS GP2 volume	IBM Spectrum Protect cloud cache disk	8
Object storage	Amazon S3 bucket	IBM Spectrum Protect Amazon S3 bucket (same AWS region as instance) Accessed via VPC endpoint using HTTPS	1

Table 4: AWS, medium configuration

Cloud component	**AWS component**	**Detailed description**	**Quantity**
Server and network	m5.12xlarge or m5a.12xlarge Amazon EC2 instance (dedicated)	48-core Intel Platinum 8175 @ 3.10 GHz or AMD EPYC 7000 @ 2.5 GHz	1

		192 GB RAM	
		EBS disk optimized with 7000 Mbps EBS throughput	
		10 Gbit Ethernet connectivity	
	Virtual Private Cloud (VPC)	Instance placed in VPC; endpoint created for Amazon S3 service in the same AWS region	1
	Operating system	Red Hat Enterprise Linux or Ubuntu Linux server	1
Block storage	100 GB EBS GP2 volume	Operating system disk	1
	100 GB EBS GP2 volume	IBM Spectrum Protect instance disk	1
	1500 GB EBS GP2 volume	IBM Spectrum Protect database disk	4
	2048 GB EBS ST1 volume	IBM Spectrum Protect database backup disk	2
	512 GB EBS GP2 volume	IBM Spectrum Protect database active log disk	1
	1024 GB EBS ST1 volume	IBM Spectrum Protect database archive log disk	3
	3300 GB EBS GP2 volume	IBM Spectrum Protect cloud cache disk	4

Object storage	Amazon S3 bucket	IBM Spectrum Protect Amazon S3 bucket (same AWS Region as instance) Accessed via VPC endpoint using HTTPS	1

Table 5: AWS, small configuration

Cloud component	**AWS component**	**Detailed description**	**Quantity**
Server and network	m5.4xlarge or m5a.4xlarge Amazon EC2 instance (shared or dedicated)	16-core Intel Platinum 8175 @ 3.10GHz or AMD EPYC 7000 @ 2.5 GHz	1
		64 GB RAM	
		EBS disk optimized with 3500 Mbps EBS throughput	
		Up to 10 Gbit Ethernet connectivity	
	Virtual Private Cloud (VPC)	Instance placed in VPC; endpoint created for Amazon S3 service in the same AWS Region	1
	Operating system	Red Hat Enterprise Linux or Ubuntu Linux server	1
Block storage	100 GB EBS GP2 volume	Operating system disk	1

	100 GB EBS GP2 volume	IBM Spectrum Protect instance disk	1
	665 GB EBS GP2 volume	IBM Spectrum Protect database disk	4
	1024 GB EBS ST1 volume	IBM Spectrum Protect database backup disk	2
	512 GB EBS GP2 volume	IBM Spectrum Protect database active log disk	1
	1024 GB EBS ST1 volume	IBM Spectrum Protect database archive log disk	1
	3300 GB EBS GP2 volume	IBM Spectrum Protect cloud cache disk	2
Object storage	Amazon S3 bucket	IBM Spectrum Protect Amazon S3 bucket (same AWS Region as instance) Accessed via VPC endpoint using HTTPS	1

Table 6: AWS, extra-small configuration

Cloud component	**AWS component**	**Detailed description**	**Quantity**
Server and network	m5.xlarge or m5a.xlarge Amazon EC2 instance (shared or dedicated)	4-core Intel Platinum 8175 @ 3.10 GHz or AMD EPYC 7000 @ 2.5 GHz	1

		16 GB RAM	
		EBS disk optimized with up to 2120 Mbps EBS throughput	
		Up to 10 Gbit Ethernet connectivity	
	Virtual Private Cloud (VPC)	Instance placed in VPC; endpoint created for Amazon S3 service in the same AWS Region	1
	Operating system	Red Hat Enterprise Linux or Ubuntu Linux server	1
Block storage	30 GB EBS GP2 volume	Operating system disk	1
	30 GB EBS GP2 volume	IBM Spectrum Protect instance disk	1
	64 GB EBS GP2 volume	IBM Spectrum Protect database and active log disk	5 (5x file systems created with Linux LVM)
	512 GB EBS ST1 volume	IBM Spectrum Protect database backup and archive log disk	1 (5x file systems created with Linux LVM)
	1200 GB EBS GP2 volume	IBM Spectrum Protect cloud cache disk	1
Object storage	Amazon S3 bucket	IBM Spectrum Protect Amazon S3 bucket (same AWS Region as instance)	1

		Accessed via VPC endpoint using HTTPS	

2.1 Design Considerations for Amazon EC2 Instances

Several factors play into the choice of instance type and disk configuration for Amazon instances. For each of the sizes, choose an instance type that will provide an acceptable balance of vCPU and memory capacity to align with IBM Spectrum Protect sizing targets. Regardless of size, any instance chosen should feature **EBS optimizations** with dedicated bandwidth to the attached disk layer. This feature is deemed necessary, given the **I/O requirements** of the IBM Spectrum Protect server database and the throughput and I/O requirements of the cloud accelerator cache as the primary backup target. Each of the preferred Amazon EC2 instances mentioned here are launched with EBS optimizations enabled by default. For instances that do not feature EBS optimizations at launch by default but that have this feature available, ensure that the EBS optimizations are enabled when configuring the instance. See References [4].

Some Amazon EC2 instance lines tend to provide too much of one resource at excess cost or too little of another resource than is needed by IBM Spectrum Protect. The other t2 and m3 “General Purpose” instance lines were not sufficient from a vCPU and memory perspective. The “Memory Optimized” x1 and r4 instances provide unnecessary amounts of memory. The “Accelerated Computing” p3, p2, g3, and f1 instance lines, geared toward machine learning and compute-intensive applications, are costly. Thus, they make a poor choice for IBM Spectrum Protect deployments

Instance lines that could potentially match IBM Spectrum Protect requirements include the **i3 and d2 lines** of “Storage Optimized” instances as well as the **c4 line** of “Compute Optimized” instances and the m4 and m5 “General Purpose” instance lines.

Of the i3 line, the i3.4xlarge instance type is a candidate for small deployments; the i3.8xlarge instance type is a candidate for medium deployments; and the i3.8xlarge and i3.16xlarge instance types are candidates for large deployments. The i3.8xlarge instance type is also capable enough from a vCPU and memory standpoint for a smaller-end large configuration. At first glance, the i3 instance line’s internal non-volatile memory express (NVMe) solid-state drive (SSD) storage could potentially be considered for use as cloud accelerator cache storage. However, it **should be cautioned** that for all non-EBS volumes, if you stop or terminate the instance, the disk volumes are erased. Even if you temporarily stop the instance with the intention of starting it later, disk volumes are erased. When the system restarts, its non-EBS volumes are provided anew, and any previous data that might have resided on disks of this type are lost. Use of this potentially **volatile storage should be avoided as storage-pool directory targets** due to the potential for data loss. Attached disks (EBS) are the preferred and only reasonable repository for IBM Spectrum Protect server data.

Of the d2 line, the d2.2xlarge instance type could serve for a small configuration, the d2.4xlarge instance type for a medium configuration, and the d2.8xlarge instance type for a

large configuration. The d2 line features internal storage that does not involve the EBS layer. The same caveats apply as mentioned for the i3 line.

Of the c4 line, the most appropriate instance type is the c4.8xlarge for small or a smaller-end medium. None of the c4 line instance types satisfy the memory requirements of a large system and should be avoided for that reason.

From a cost point of view, the m5.2xlarge instance type for small, the m5.4xlarge instance type for medium, and the m5.10xlarge instance type for large are each much less expensive than their i3, d2, or c4 counterparts in terms of hourly "On Demand" runtime cost. However, of the m5 line, the **m5.16xlarge** instance type is best suited for a true large system given its vCPU and memory capacity. For this reason, the m5.16xlarge instance type is the current preferred choice.

In general, the current generation of the **m5 line of Amazon EC2 instances is preferred** for large, medium, small, and extra-small IBM Spectrum Protect deployments. This newer "general purpose" instance type provides a good combination of CPU, memory, Ethernet network, and EBS disk performance with the latest Intel Platinum 8175 or AMD EPYC 7000 series processors. In addition, the m5 instances are typically more affordable than their earlier generation m4 counterparts. Internal lab benchmarks have found that AMD EPYC-based m5a instances provide better overall performance at scale with IBM Spectrum Protect than Intel Platinum models, at a slightly lower cost.

If you choose an instance type that does **not meet** the memory requirements of a Blueprint-level system, mitigation steps might be necessary to avoid out-of-memory conditions when the system is under a Blueprint-level load. Mitigation steps can include use of the `DBMEMPERCENT` server parameter to tune down maximum system Db2 database memory at the expense of database performance, reducing the maximum number of parallel client sessions dispatched to the server, or both. You might apply additional measures. If in doubt, adhere to the physical Blueprint targets.

Table 7: AWS instance options

Sizing category	Instance line	Instance type	vCPU	Memory	Blueprint CPU	Blue-print memory
Extra small	m5(a)	M5(a).xlarge	4	16 GB	-	-
Small	m5(a)	m5(a)xlarge	16	64 GB	12	64 GB
	i3	i3.4xlarge	16	122 GB		
	d2	d2.2xlarge	8	61 GB		
Medium	c4	c4.8xlarge	36	60 GB	16	128 GB
	m5(a)	m5(a).12xlarge	48	192 GB		
	i3	i3.8xlarge	32	244 GB		
	d2	d2.4xlarge	16	122 GB		
Large	m4	m4.10xlarge	40	160 GB	44	

	i3	i3.8xlarge	32	244 GB		256 GB (V3.1) or 384 GB (V4.1)
	m5(a)	m5(a).16xlarge	64	256 GB		
	i3	i3.16xlarge	64	488 GB		
	d2	d2.8xlarge	36	244 GB		

The previous table is sorted from **lowest to highest** cost instance in each category. Costs are estimated from the Amazon EC2 calculator based on a Red Hat Enterprise Linux instance in the US West Oregon Region with On-Demand hourly pricing. Pricing is subject to change; for this reason, exact figures are not provided. The entries highlighted in blue are the current preferred Amazon EC2 instance types for extra-small, small, medium, and large configurations, respectively. The blue entries provide a good balance between controlling costs while satisfying IBM Spectrum Protect server requirements.

Beginning with IBM Spectrum Protect V8.1.3, the server automatically throttles client backup operations if the cloud accelerator cache portion of a direct-to-cloud storage pool is nearing full capacity. As a result, it is not mandatory to configure cloud accelerator disk cache space that would be large enough to hold a full day's worth of backups (after data deduplication and compression). However, disk benchmarks should be run to ensure that the anticipated back-end workload that an IBM Spectrum Protect server is expected to support will not result in this disk location being the primary bottleneck of the system (see Disk Benchmarking). In the previous table, the EC2 instances were carefully selected to ensure that they have the dedicated EBS bandwidth and disk resources that are necessary for the desired role. In practice, any planned deployment should be validated to ensure it will meet performance requirements.

The NVMe SSD storage available for many instances, including the i3 line, should be avoided for use by IBM Spectrum Protect. If the instance is stopped, all data in this non-EBS disk will be lost. This would include a scenario in which data that is ingested by the IBM Spectrum Protect server into the cloud accelerator cache, and reported successfully stored to the client, is lost from this disk before the asynchronous disk-to-cloud transfer is completed.

Thresholds, particularly at the EBS disk level, must be kept in mind when designing and deploying complete instance solutions. For example, IOPS are limited to a count of 80,000 per AWS account per disk type. Failure to account for IOPS or throughput limitations imposed by the service provider on the account, instance, or disk level can result in performance problems that might be difficult to debug in production. For Amazon documentation about this subject, see References [4] [5].

2.1.1 Considerations for Direct-to-Cloud Architectures

The **direct-to-cloud** configurations discussed here are architected such that the disk of the cloud accelerator cache is on fast-performing SSD. In general, the limiting factor for end-to-end ingestion throughput for an IBM Spectrum Protect server using object storage is the network to the object storage system or the object storage system itself. For this reason, you must understand the throughput capability requirements of the planned system. Those

requirements will drive your decisions about assigning servers to Ethernet networks. Next, the ceiling for daily ingestion throughput (in terms of mebibytes per second or tebibytes per day) must be determined at the object storage end by using **object storage benchmarking** tests. The test results will establish what the overlapped I/O capability would have to be for the cloud accelerator cache disk location for maximum back-end throughput. These two factors will help you select a disk technology to sustain daily backup ingestion requirements while working within object storage limits. After you select a disk technology, run **disk benchmarking** tests to ensure throughput capability (see Disk Benchmarking).

Tip: In this paper, the abbreviation MiB is used for mebibytes, the abbreviation TiB is used for tebibytes, the abbreviation KiB is used for kibibytes, and the abbreviation GiB is used for gibibytes.

Example

If an object storage link is capable of 10 Gbps, this data transfer speed equals about 1000 MiB/s after packet overhead and other efficiency loss. In order to saturate this link for long periods, the cloud accelerator cache disk location must be capable of taking in client ingestion data (writes) at 1000 MiB/s and transmitting staged data to object storage (reads) at a similar speed, 1000 MiB/s (~128-256 KiB I/O size). This capacity ensures that the cloud accelerator cache disk can remain as small as possible while sustaining maximum throughput. **Alternatively**, a larger capacity, slower disk technology (such as Amazon EBS st1 magnetic disk) can be used such that the client ingestion data that has been staged to accelerator disk cache can be transmitted to object storage over a longer period of the day (extending past the backup window). However, beware that data residing only in the cloud accelerator cache is unprotected in the sense that only a single copy of the data exists. The redundancy protection inherent in cloud object storage is available only if the data is transmitted to object storage. Generally, Amazon EBS disks provide acceptable durability.

2.1.2 Sizing the Cloud Accelerator Cache

Figure 4 can be used as a rough guide for the appropriate disk technology to use based on object storage and object storage network capability. At the top left, Amazon S3 object storage is reachable over the same LAN (for example, within the same AWS Virtual Private Cloud network). As we move from top to bottom in the diagram, the network capability becomes slower (10 Gbps to 1 Gbps), while the storage capacity requirements increase to store data that is queued up in the accelerator disk cache awaiting transfer to object storage. In the case of slower network-to-object storage, it is more likely that (asynchronous) client ingestion from local client systems can run at a faster rate than cloud transfer. In such a scenario, client ingestion data begins to fill the cache disk location faster than the data can be transferred to object storage and cleared from the cache. As of IBM Spectrum Protect V8.1.2, an internal **throttling** mechanism is in place to slow client ingestion speeds if the cloud accelerator cache disk area begins nearing capacity. However, to avoid slowing client ingestion in cases where ingestion exceeds the cloud transfer rate (which might not be desired), the accelerator cache should be sized with a larger capacity, perhaps up to a single day's worth of back-end client ingestion (after data deduplication and compression).

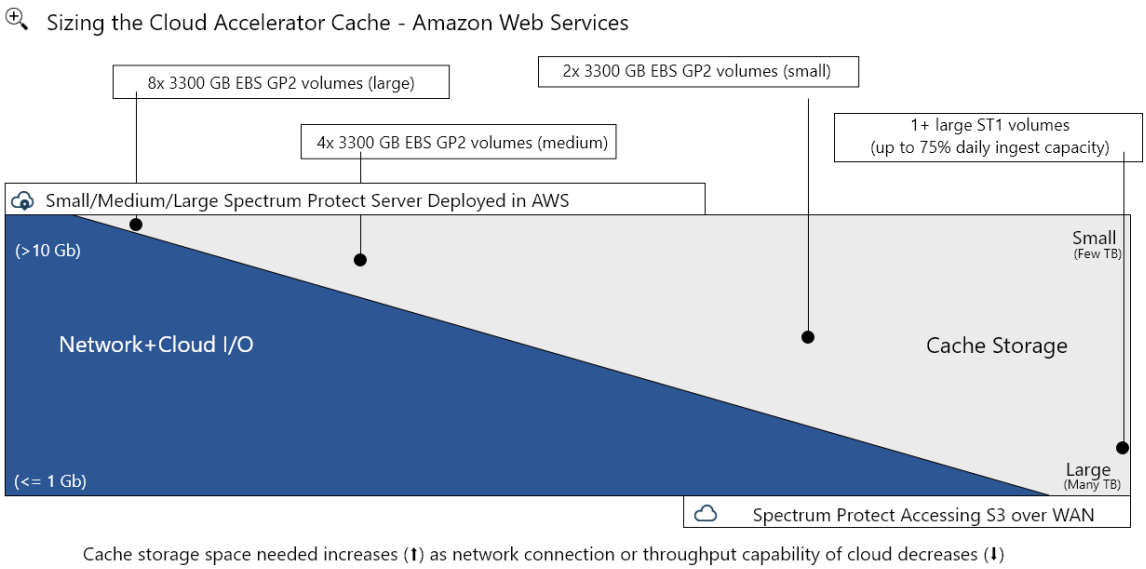


Figure 4: Sizing the cloud accelerator cache for AWS

2.1.3 AWS: Large Instance Considerations

For a large Amazon based build, the m5(a).16xlarge instance type is the preferred choice. The newer generation m5 or m5a “general purpose” instance lines feature updated Intel Platinum or AMD EPYC processors on the AWS Nitro System hypervisor and are generally more affordable than their earlier m4 counterparts. The AWS Nitro System instances are generally higher-performing instance types with lines including m5, c5, and r5 instance types. For more information, see References [9].

Amazon **GP2 type EBS disks** were chosen for the IBM Spectrum Protect **database disk** and active log disk volumes as a more cost-effective alternative to io1 type disks. To meet IOPS requirements of the Db2 component, a larger capacity disk is required to enable the increased IOPS capability of these underlying “general purpose” EBS disks. Eight disks at 2000 GB each for the database volumes and one disk at 665 GB for the active log were sufficient for this purpose. A capacity of approximately 15.6 TB represents an overprovisioning of space for the database in exchange for the required IOPS scalability.

Amazon **st1 type** throughput-optimized EBS disks were chosen for use as IBM Spectrum Protect **database backup** and **database archive log** volumes due to their sequentially optimized nature and cost effectiveness.

Amazon **gp2 type EBS disks** were chosen for **cloud accelerator cache** due to their efficient random IOPS characteristics. Configurations with these types of disks seem to be ideal for handling the direct-to-cloud pattern of 256 KB sequential writes mixed with asynchronous 128 KB, somewhat sequential reads (of other files on disk). Eight of these disks at a 3300 GB capacity each were chosen to provide enough overlapped throughput capability to allow the IBM Spectrum Protect server to saturate the back-end link to S3 storage from this instance. A total capacity of more than 25 TB results from this configuration, providing enough storage for a single day’s worth of large Blueprint-level

backup ingestion, assuming a favorable (for example, 4:1) data reduction ratio with deduplication and compression of front-end client data.

To review disk benchmarks for database disk volumes and cloud accelerator cache volumes for all built configurations, see Disk Benchmarking.

2.1.4 AWS: Medium Instance Considerations

For the medium Amazon build, the m5(a).12xlarge instance types are preferred (Intel or AMD variants). This instance type provides 48 vCPUs and 192 GB of RAM with 875 MB/s of advertised dedicated EBS throughput and 10 Gbit Ethernet network capability. This instance provides for more CPU and memory capacity than would typically be required of a medium Blueprint-level system, allowing this instance type to be an option for a smaller-end large build as well. Other alternatives for the medium Blueprint role tend to provide less memory than is preferred, and this factor could lead to resource constraints on a loaded system in production. More cost-effective alternatives to the medium role could be pursued within EC2, with the caveat that **memory demands** and usage of the IBM Spectrum Protect server and database should be closely monitored during high load.

As with the large instance type, Amazon gp2 disks were again chosen for the primary database volumes and active log volume, but at a smaller capacity for both. A total database size of approximately 5.8 TB is provided with this configuration. Once again, excess capacity is provisioned with these less expensive disks to achieve the desired IOPS capability. st1 disks were again chosen for the server's database backup and archive log disk volumes.

For the cloud accelerator cache disk, gp2 volumes were again chosen. However, only four are needed for a medium configuration.

2.1.5 AWS: Small Instance Considerations

For the small Amazon build, the m5(a).4xlarge instance types are preferred (Intel or AMD variants). This instance type provides 16 vCPUs and 122 GB of RAM, providing more server memory than the physical Blueprint specification. This instance type provides for 437.5 MB/s of dedicated throughput to the EBS disk layer, along with up to 10 Gbit Ethernet bandwidth capability. A similar caveat applies with this instance as with the medium EC2 instance: options are present that could serve the role of an IBM Spectrum Protect medium deployment but with less memory capability than is preferred, at the risk of being memory constrained on the system during periods of high load.

Again, gp2 type EBS disks were used here for database and active log volumes, providing 2.6 TB of database capacity across four database disks. Disk capacities for the st1-based database backup and archive log volumes were also reduced.

In this case, only two 3300 GB gp2 type volumes were used for cloud accelerator cache, providing a total capacity of 6600 GB.

2.1.6 AWS: Extra-Small Instance Considerations

For the extra-small Amazon build, the m5(a).xlarge instance types are preferred (Intel or AMD variants). This instance type features 4 vCPUs and 64 GB of RAM. For maximum cost savings, the m5a.xlarge AMD variant and Ubuntu Linux could be chosen. This

instance type provides up to 2120 Mbps of dedicated throughput to the EBS disk layer, along with up to 10 Gbit Ethernet bandwidth.

EBS gp2 disks were again chosen for the database and active log volumes to ensure adequate IOPs capability, even for this smaller instance size. For convenience with scalability, the five 64 GB disks were combined into a single LVM volume group with four database volumes and one active log volume created. Similarly, a single 512 GB ST1 volume was provisioned and divided into two volumes using LVM, one for database backup and one for use as the archive log volume. If additional database backup capacity is required, additional EBS disk capacity could be added to this volume group. This applies as well to the one 1200 GB GP2 disk provisioned for the cloud accelerator cache. General purpose SSD performance is still preferable for the overlapped I/O seen with this volume.

Throughput Measurements and Results

Throughput measurements for backup and restore operations are provided with representative datasets for selected configurations. All throughput measurements were collected with IBM Spectrum Protect servers configured in a direct-to-cloud, accelerator cache-backed setup, with primary backup data stored to object storage. Throughput measurements with a disk-to-cloud tiering configuration are not included in this document.

For each configuration tested, the following preferred settings were adopted for the cloud-container storage pool. To optimize performance when you deploy an IBM Spectrum Protect server instance in the cloud, use the following settings:

- Storage pool compression enabled
- Storage pool encryption enabled
- Storage pool configured as an off-premises cloud-container storage pool
- HTTPS URLs specified for all cloud endpoints

3.1 Dataset Descriptions

For the performance tests that were conducted within these environments, all or a subset of the following datasets were used:

Table 8: Throughput measurement datasets

Front-end object size	Average duplication extent size	Duplicate data percentage	Data compressibility	Notes
128 MiByte	~150-200 KiByte	~70%	~50%	VE-like, favorable extent size
128 MiByte	~200-300 KiByte	~0%	~0%	Random data, large extent size

1 GiByte	~60-100 KiByte	~50%	~50%	DB-like, small extent size
128 KiByte	~128 KiByte	~0%	~95%	Small file, high overhead

The **128 MiByte, VE-like** front-end dataset represents a relatively large object size that aligns with the IBM Spectrum Protect for Virtual Environments: Data Protection for VMware API client's **VE megablock** size for virtual machine disk backups. The large object size and **relatively large, though realistic, deduplication extent** size represents a favorable profile for the IBM Spectrum Protect server's ingestion engine to achieve good performance. A duplicate data rate of 70% combined with a compressibility rate of 50% for this dataset yields an **85% total data reduction** from front-end data as compared with data that is actually stored to the (cloud-accelerator cache and object storage) back-end after data deduplication, compression, and encryption processing. Although this workload does not qualify as a "best case," it does represent a realistic, favorable scenario in which to model top-end throughput capability of an IBM Spectrum Protect system without overestimating throughput performance.

The **128 MiByte, random** front-end dataset represents a larger object size with a large, favorable deduplication extent size. However, the random nature of the data ensures that it does not deduplicate well with existing storage pool data or compress well. This dataset is included to represent a workload that is throughput intensive from the perspective of storage pool disk and object storage network load. Full backups of large objects containing relatively random data content would be modeled well by this dataset.

The **1 GiByte** front-end dataset represents a model of **structured, database-like data** possessing a relatively small deduplication extent size relative to the front-end object size. Such a workload is representative of what might be experienced with an IBM Spectrum Protect for Databases: Data Protection for Oracle backup environment protecting production databases. The smaller extent size causes additional strain and overhead for the IBM Spectrum Protect ingestion engine and typically results in less throughput than the 128 MiByte dataset. A duplicate data rate of 50% and compressibility of 50% yield a 75% overall front-end to back-end reduction for this workload, with a **4:1 ratio** reduction, which approaches what is seen for this type of data in the field.

The **128 KiByte** front-end dataset is used here as a relative "worst case" workload in regard to throughput efficiency for the IBM Spectrum Protect server. This **"small file" workload** will stress the data deduplication, compression, encryption, and object storage transfer components of the IBM Spectrum Protect server to a higher degree relative to the amount of data that is actually protected and transferred to object storage. This high overhead dataset allows for predicting a lower estimate on performance of an IBM Spectrum Protect environment within these cloud solutions.

3.2 Backup and Restore Measurements

The following sections outline the backup and restore throughput results that were experienced with the previously mentioned datasets in the built cloud environments.

Prior to conducting backup and restore tests on the IBM Spectrum protect environments, a **load phase** was conducted whereby the servers were initially loaded with a set of deduplicated 128 MiByte front-end data in order to populate the server database tables to provide for a more realistic customer configuration. IBM Spectrum Protect database queries can change their behavior based on the size and layout of server database tables. This load phase was performed to bring behavior in line with real environment expectations.

For each dataset, up to 50 IBM Spectrum Protect client backup sessions were initiated in parallel to the server for the large Amazon configuration, up to 25 for the medium Amazon configuration, and up to 10 for the small Amazon configuration. The results presented here for backup represent the maximum front-end throughput experienced with the largest number of sessions tested against that system.

For each dataset on restore, between 1 and 40 client restore sessions were initiated for the large Amazon system, between 1 and 20 for the medium Amazon system, and between 1 and 10 for the small Amazon system. Results presented here include the intermediate session count values to show how restore throughput can scale with the number of restore sessions involved for datasets similar to these types.

All throughput values represent front-end, "protected data" values, before inline data deduplication, compression, and encryption. These are the data rates experienced by a **client** that is backing up data to or restoring data from the IBM Spectrum Protect server. The rates are similar to what customers would likely describe as their performance experience with the product. On ingestion, the actual quantity of data that makes it to accelerator cache disk and onwards to object storage will be less, depending on the data deduplication and compression rate. On restore, all individual extents comprising a front-end object will be restored using HTTP GET calls from the object storage device. However, the built-in caching within the IBM Spectrum Protect server's restore engine might reduce the number of restore operations needed if a workload contains duplicate data.

3.2.1 AWS: Large Instance Measurements

128 MiByte VE-like front-end dataset results:

Table 9: AWS, large configuration, 128 MiByte VE-like dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
50	**70%**	**2147.8**	**7.4**	**59.0**	**73.7**

For the large Amazon build, front-end aggregate throughput for the 50-session favorable dataset achieved 6.9 TiBytes/hour, which would yield capability well within the large Blueprint range. Values are provided here for estimates using both 8-hour and 10-hour backup windows in a given day. Both estimates are provided for convenience to align with our experience that customers generally use one of these two time periods for their daily ingestion workloads.

Table 10: AWS, large configuration, 128 MiByte VE-like dataset restore

Sessions	MiBytes/s	Gibytes/hour
1	209.7	737.1
2	392.3	1379.0
4	627.7	2206.9
8	1107.1	3892.3
16	1128.4	3966.9
32	1736.8	6105.9

With the 50% compressibility of this dataset, the amount of data transmitted over the wire (from object storage) is half its decompressed front-end value. This allows for throughput in excess of 10 Gbit line speed without saturating the EC2-to-S3 Ethernet link. Restore rates from object storage are typically challenged due to latency and individual read characteristics. Restore throughput rates with realistic session counts from object storage pools might challenge certain data recovery time objectives (RTOs).

128 MiByte random front-end dataset results:

Table 11: AWS, large configuration, 128 MiByte random dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
50	**0%**	**296.2**	**1.0**	**8.1**	**10.2**

Table 12: AWS, large configuration, 128 MiByte random dataset restore

Sessions	MiBytes/s	Gibytes/hour
1	145.0	509.8
2	253.8	892.4
4	385.4	1354.9
8	477.9	1680.0
16	540.2	1899.1
32	405.2	1424.7

This dataset benefits from a large deduplication extent size of approximately 200 - 300 KiBytes, on average. This allows for more data movement per HTTP GET operation performed by the cloud-container storage pool and subsequently greater throughput per session compared to the previous 128 MiByte dataset.

1 GiByte front-end dataset results:

Table 13: AWS, large configuration, 1 GiByte dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
50	**50%**	**1538.1**	**5.3**	**42.2**	**52.8**

Ingestion throughput rates with the 1 GiByte dataset featuring the smaller (60 – 100 KiByte) average deduplication extent size were less than the 128 MiByte favorable workload, but still in line with large Blueprint throughput targets in this direct-to-cloud configuration.

Table 14: AWS, large configuration, 1 GiByte dataset restore results

Sessions	MiBytes/s	Gibytes/hour
1	129.9	456.7
2	235.3	827.4
4	291.4	1024.3

8	446.0	1568.1
16	346.9	1219.5
32	477.4	1678.5

Restore throughput rates with the smaller extent workload are somewhat less than the favorable, 128 MiByte dataset. This is due to the additional overhead of more HTTP GET requests necessary to restore this data (smaller extents lead to more operations per front-end restored data). A throughput of more than 1 TiByte/hour could be achieved with 20-40 restore sessions of this data type.

3.2.2 AWS: Medium Instance Measurements

128 MiByte VE-like front-end dataset results:

Table 15: AWS, medium configuration, 128 MiByte VE-like dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
25	**70%**	**1108.9**	**3.8**	**30.5**	**38.1**

Throughput with the favorable 128 MiByte object dataset with the medium Amazon build easily falls within the upper end of the medium Blueprint throughput target with 25 backup sessions for an 8-hour backup window and surpasses it with a 10-hour window. This configuration could possibly support the data workload of a smaller, large Blueprint role.

Table 16: AWS, medium configuration, 128 MiByte VE-like dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	254.4	894.5
2	506.4	1780.4
4	859.3	3020.9
8	1823.2	6409.6
16	1910.4	6716.2

128 MiByte random front-end dataset results:

Table 17: AWS, medium configuration, 128 MiByte random dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
25	**0%**	**207.0**	**0.7**	**5.7**	**7.1**

Table 18: AWS, medium configuration, 128 MiByte random dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	163.7	575.4
2	268.3	943.2
4	419.1	1473.5
8	444.1	1561.2
16	463.8	1630.5

1 GiByte front-end dataset results:

Table 19: AWS, medium configuration, 1 GiByte dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
25	**50%**	**775.8**	**2.7**	**21.3**	**26.6**

With the 1 GiByte dataset workload, ingestion throughput for 8- or 10-hour backup windows is still within medium Blueprint targets (6 – 20 TiBytes per day front-end data protected).

Table 20: AWS, medium configuration, 1 GiByte dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	157.4	553.2

2	313.1	1100.7
4	644.4	2265.5
8	593.0	2084.7
16	578.8	2034.8

3.2.3 AWS: Small Instance Measurements

128 MiByte VE-like front-end dataset results:

Table 21: AWS, small configuration, 128 MiByte VE-like dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
10	**70%**	**825.8**	**2.8**	**22.7**	**28.4**

Table 22: AWS, small configuration, 128 MiByte VE-like dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	226.2	795.1
2	415.2	1459.6
4	644.1	2264.5
8	887.9	3121.6

128 MiByte random front-end dataset results:

Table 23: AWS, small configuration, 128 MiByte random dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
10	**0%**	**104.1**	**0.4**	**2.9**	**3.6**

Table 24: AWS, small configuration, 128 MiByte random dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	141.0	495.7
2	223.4	785.4
4	286.1	1005.9
8	339.5	1193.7

1 GiByte front-end dataset results:

Table 25: AWS, small configuration, 1 GiByte dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
10	**50%**	**808.6**	**2.8**	**22.2**	**27.8**

Table 26: AWS, small configuration, 1 GiByte dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	140.1	492.7
2	244.1	858.1
4	336.0	1181.3
8	391.3	1375.6

3.2.4 AWS: Extra-Small Instance Measurements

128 MiByte VE-like front-end dataset results:

Table 27: AWS, extra-small configuration, 128 MiByte VE-like dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
5	**70%**	**306.9**	**1.1**	**8.4**	**10.5**

Table 28: AWS, extra-small configuration, 128 MiByte VE-like dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	180.1	633.2
2	330.7	1162.6
4	555.4	1952.6

128 MiByte random front-end dataset results:

Table 29: AWS, extra-small configuration, 128 MiByte random dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
5	**0%**	**50.0**	**0.2**	**1.4**	**1.7**

Table 30: AWS, extra-small configuration, 128 MiByte random dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	96.7	339.8
2	109.5	384.8
4	122.1	429.1

1 GiByte front-end dataset results:

Table 31: AWS, extra-small configuration, 1 GiByte dataset backup results

Sessions	Data deduplication %	MiBytes/s	TiBytes/hour	TiBytes per 8 hours	TiBytes per 10 hours
5	**50%**	**267.1**	**0.9**	**7.3**	**9.2**

Table 32: AWS, extra-small configuration, 1 GiByte dataset restore results

Sessions	MiBytes/s	GiBytes/hour
1	120.4	423.4
2	172.2	605.4
4	190.6	670.1

APPENDIX

Disk Setup Commands for Linux Deployments

For IBM Spectrum Protect deployments on AWS cloud computing systems, the preferred operating system is Linux, either the latest IBM Spectrum Protect supported Red Hat Enterprise Linux or Ubuntu Linux. With Ubuntu Linux, care should be taken to ensure that all required Linux packages are installed to enable Linux Logical Volume Manager (LVM) functionality. For more information about the operating systems, see the IBM Spectrum Protect technote (References [7]).

This chapter provides reference information for deployments of IBM Spectrum Protect with Ubuntu Linux 16.04 LTS for large, medium, small, and extra-small environments. In this

case, Ubuntu Linux was installed on each instance on the Amazon EC2 cloud computing system. The following sections outline the disk, file system, and miscellaneous commands that were executed after launching the instance in preparation for installing IBM Spectrum Protect. These sections reflect the preferred instance type and direct-to-cloud disk configurations that are outlined in Amazon EC2 Compute Configurations.

If you use Linux LVM and logical volumes for the cloud accelerator cache (and sometimes archive log) disk, ensure that **LVM striping** is used with the `lvcreate` command such that logical volumes make use of data stripes from all physical disks in the volume group. The number of stripes that you specify when creating a logical volume should match the number of physical disks in the volume group. Lab testing indicates that a suitable stripe size for cloud accelerator cache and archive log purposes is **16 KiBytes**. For example, for a cloud accelerator cache volume group making use of two physical disks, you could use the following command:

```
lvcreate --stripes 2 --stripesize 16 --extents 100%FREE --name sp_cc_vg sp_cc
```

IBM Spectrum Protect, Large Amazon EC2 System, Ubuntu Linux: m5a.16xlarge Instance Type

```
############################################
#   Miscellaneous
############################################

# Update apt packages, install tooling
apt update
apt upgrade -y
apt install scsitools -y
apt install dstat -y
apt install sysstat -y
apt install multipath-tools -y
apt install multipath-tools-boot -y
# Start the multipath daemon
touch /etc/multipath.conf
systemctl restart multipath-tools.service

# Set server hostname
hostnamectl set-hostname <desired hostname>

# Set vm.swappiness to 5 to match v4.1 xLinux blueprint recommendation
sysctl vm.swappiness=5
```

```
# Install the Korn shell
apt install ksh -y
# Reboot the system


##############################################
#   Spectrum Protect Disk Setup (Ubuntu)
##############################################

# Run the following as the "root" user

# Install LVM for Ubuntu
apt install lvm2 -y
systemctl enable lvm2-lvmetad
systemctl start lvm2-lvmetad
# Make the required Spectrum Protect directories
mkdir /sp
mkdir /sp/tsminst1
mkdir /sp/sp_db1
mkdir /sp/sp_db2
mkdir /sp/sp_db3
mkdir /sp/sp_db4
mkdir /sp/sp_db5
mkdir /sp/sp_db6
mkdir /sp/sp_db7
mkdir /sp/sp_db8
mkdir /sp/sp_dbb1
mkdir /sp/sp_dbb2
mkdir /sp/sp_alog
mkdir /sp/sp_archlog
mkdir /sp/sp_cc

# Note that disk name values may be different from those seen here in actual
deployments

# Instance directory
# Change the value for "nvme*" to match your environment
mkfs.xfs /dev/nvme8n1
mount /dev/nvme8n1 /sp/tsminst1
```

```
# Cloud Cache
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme4n1
pvcreate /dev/nvme6n1
pvcreate /dev/nvme7n1
pvcreate /dev/nvme10n1
pvcreate /dev/nvme12n1
pvcreate /dev/nvme14n1
pvcreate /dev/nvme18n1
pvcreate /dev/nvme19n1
vgcreate sp_cc /dev/nvme4n1 /dev/nvme6n1 /dev/nvme7n1 /dev/nvme10n1 /dev/nvme12n1
/dev/nvme14n1 /dev/nvme18n1 /dev/nvme19n1
lvcreate --stripes 8 --stripesize 16 --extents 100%FREE --name sp_cc_vg sp_cc
mkfs.xfs /dev/mapper/sp_cc-sp_cc_vg
mount /dev/mapper/sp_cc-sp_cc_vg /sp/sp_cc


# DB
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme1n1
pvcreate /dev/nvme3n1
pvcreate /dev/nvme5n1
pvcreate /dev/nvme9n1
pvcreate /dev/nvme11n1
pvcreate /dev/nvme15n1
pvcreate /dev/nvme16n1
pvcreate /dev/nvme22n1
vgcreate sp_db /dev/nvme1n1 /dev/nvme3n1 /dev/nvme5n1 /dev/nvme9n1 /dev/nvme11n1
/dev/nvme15n1 /dev/nvme16n1 /dev/nvme22n1
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db1 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db2 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db3 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db4 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db5 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db6 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db7 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 511999 --name sp_db_db8 sp_db
mkfs.ext4 /dev/mapper/sp_db-sp_db_db1
mkfs.ext4 /dev/mapper/sp_db-sp_db_db2
```

```
mkfs.ext4 /dev/mapper/sp_db-sp_db_db3
mkfs.ext4 /dev/mapper/sp_db-sp_db_db4
mkfs.ext4 /dev/mapper/sp_db-sp_db_db5
mkfs.ext4 /dev/mapper/sp_db-sp_db_db6
mkfs.ext4 /dev/mapper/sp_db-sp_db_db7
mkfs.ext4 /dev/mapper/sp_db-sp_db_db8
mount /dev/mapper/sp_db-sp_db_db1 /sp/sp_db1
mount /dev/mapper/sp_db-sp_db_db2 /sp/sp_db2
mount /dev/mapper/sp_db-sp_db_db3 /sp/sp_db3
mount /dev/mapper/sp_db-sp_db_db4 /sp/sp_db4
mount /dev/mapper/sp_db-sp_db_db5 /sp/sp_db5
mount /dev/mapper/sp_db-sp_db_db6 /sp/sp_db6
mount /dev/mapper/sp_db-sp_db_db7 /sp/sp_db7
mount /dev/mapper/sp_db-sp_db_db8 /sp/sp_db8

# Active Log
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme21n1
mount /dev/nvme21n1 /sp/sp_alog

# DB backup
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme13n1
mkfs.ext4 /dev/nvme20n1
mount /dev/nvme13n1 /sp/sp_dbb1
mount /dev/nvme20n1 /sp/sp_dbb2

# Archive log
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme2n1
pvcreate /dev/nvme17n1
vgcreate sp_archlog /dev/nvme2n1 /dev/nvme17n1
lvcreate --stripes 2 --stripesize 16 --extents 100%FREE --name sp_archlog_vg
sp_archlog
mkfs.ext4 /dev/mapper/sp_archlog-sp_archlog_vg
mount /dev/mapper/sp_archlog-sp_archlog_vg /sp/sp_archlog

# Edit /etc/fstab to ensure that mount points will be restored on reboot
vi /etc/fstab
```

```
##############################################
#   Spectrum Protect Instance User Setup
##############################################

# Run the following as the "root" user.
# Add a Linux group and user to own the Spectrum Protect instance.
# Set the password for this instance.
groupadd tsmsrvrs
useradd -g tsmsrvrs tsminst1
passwd tsminst1

# Ensure that this user can use the mount points previously created
chown -R tsminst1:tsmsrvrs /sp/

# From here, proceed with the IBM Spectrum Protect installation and setup
```

IBM Spectrum Protect, Medium Amazon EC2 System, Ubuntu Linux: m5a.12xlarge Instance Type

```
##############################################
#   Miscellaneous
##############################################

# Update apt packages, install tooling
apt update
apt upgrade -y
apt install scsitools -y
apt install dstat -y
apt install sysstat -y
apt install multipath-tools -y
apt install multipath-tools-boot -y
# Start the multipath daemon
touch /etc/multipath.conf
systemctl restart multipath-tools.service
```

```
# Set server hostname
hostnamectl set-hostname <desired hostname>

# Set vm.swappiness to 5 to match v4.1 xLinux blueprint recommendation
sysctl vm.swappiness=5

# Install the Korn shell
apt install ksh -y
# Reboot the system



##############################################
#   Spectrum Protect Disk Setup (Ubuntu)
##############################################

# Run the following as the "root" user

# Install LVM for Ubuntu
apt install lvm2 -y
systemctl enable lvm2-lvmetad
systemctl start lvm2-lvmetad
# Make the required Spectrum Protect directories
mkdir /sp
mkdir /sp/tsminst1
mkdir /sp/sp_db1
mkdir /sp/sp_db2
mkdir /sp/sp_db3
mkdir /sp/sp_db4
mkdir /sp/sp_dbb1
mkdir /sp/sp_dbb2
mkdir /sp/sp_alog
mkdir /sp/sp_archlog
mkdir /sp/sp_cc

# Note that disk name values may be different from those seen here in actual
deployments

# Instance directory
# Change the value for "nvme*" to match your environment
```

```
mkfs.xfs /dev/nvme6n1
mount /dev/nvme6n1 /sp/tsminst1

# Cloud Cache
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme2n1
pvcreate /dev/nvme5n1
pvcreate /dev/nvme9n1
pvcreate /dev/nvme14n1
vgcreate sp_cc /dev/nvme2n1 /dev/nvme5n1 /dev/nvme9n1 /dev/nvme14n1
lvcreate --stripes 4 --stripesize 16 --extents 100%FREE --name sp_cc_vg sp_cc
mkfs.xfs /dev/mapper/sp_cc-sp_cc_vg
mount /dev/mapper/sp_cc-sp_cc_vg /sp/sp_cc

# DB
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme8n1
pvcreate /dev/nvme11n1
pvcreate /dev/nvme12n1
pvcreate /dev/nvme15n1
vgcreate sp_db /dev/nvme8n1 /dev/nvme11n1 /dev/nvme12n1 /dev/nvme15n1
lvcreate --stripes 4 --stripesize 16 --extents 383999 --name sp_db_db1 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 383999 --name sp_db_db2 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 383999 --name sp_db_db3 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 383996 --name sp_db_db4 sp_db
mkfs.ext4 /dev/mapper/sp_db-sp_db_db1
mkfs.ext4 /dev/mapper/sp_db-sp_db_db2
mkfs.ext4 /dev/mapper/sp_db-sp_db_db3
mkfs.ext4 /dev/mapper/sp_db-sp_db_db4
mount /dev/mapper/sp_db-sp_db_db1 /sp/sp_db1
mount /dev/mapper/sp_db-sp_db_db2 /sp/sp_db2
mount /dev/mapper/sp_db-sp_db_db3 /sp/sp_db3
mount /dev/mapper/sp_db-sp_db_db4 /sp/sp_db4

# Active Log
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme3n1
mount /dev/nvme3n1 /sp/sp_alog
```

```
# DB backup
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme4n1
mkfs.ext4 /dev/nvme13n1
mount /dev/nvme4n1 /sp/sp_dbb1
mount /dev/nvme13n1 /sp/sp_dbb2

# Archive log
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme1n1
pvcreate /dev/nvme7n1
pvcreate /dev/nvme10n1
vgcreate sp_archlog /dev/nvme1n1 /dev/nvme7n1 /dev/nvme10n1
lvcreate --stripes 3 --stripesize 16 --extents 100%FREE --name sp_archlog_vg
sp_archlog
mkfs.ext4 /dev/mapper/sp_archlog-sp_archlog_vg
mount /dev/mapper/sp_archlog-sp_archlog_vg /sp/sp_archlog

# Edit /etc/fstab to ensure that mount points will be restored on reboot
vi /etc/fstab



##############################################
#   Spectrum Protect Instance User Setup
##############################################

# Run the following as the "root" user.
# Add a Linux group and user to own the Spectrum Protect instance.
# Set the password for this instance.
groupadd tsmsrvrs
useradd -g tsmsrvrs tsminst1
passwd tsminst1

# Ensure that this user can use the mount points previously created
chown -R tsminst1:tsmsrvrs /sp/

# From here, proceed with the IBM Spectrum Protect installation and setup
```

IBM Spectrum Protect, Small Amazon EC2 System, Ubuntu Linux: m5a.4xlarge Instance Type

```
############################################
#   Miscellaneous
############################################

# Update apt packages, install tooling
apt update
apt upgrade -y
apt install scsitools -y
apt install dstat -y
apt install sysstat -y
apt install multipath-tools -y
apt install multipath-tools-boot -y

# Start the multipath daemon
touch /etc/multipath.conf
systemctl restart multipath-tools.service
# Set server hostname
hostnamectl set-hostname <desired hostname>

# Set vm.swappiness to 5 to match v4.1 xLinux blueprint recommendation
sysctl vm.swappiness=5

# Install the Korn shell
apt install ksh -y
# Reboot the system

############################################
#   Spectrum Protect Disk Setup (Ubuntu)
############################################

# Run the following as the "root" user

# Install LVM for Ubuntu
apt install lvm2 -y
systemctl enable lvm2-lvmetad
```

```
systemctl start lvm2-lvmetad
# Make the required Spectrum Protect directories
mkdir /sp
mkdir /sp/tsminst1
mkdir /sp/sp_db1
mkdir /sp/sp_db2
mkdir /sp/sp_db3
mkdir /sp/sp_db4
mkdir /sp/sp_dbb1
mkdir /sp/sp_dbb2
mkdir /sp/sp_alog
mkdir /sp/sp_archlog
mkdir /sp/sp_cc

# Note that disk name values may be different from those seen here in actual
deployments

# Instance directory
# Change the value for "nvme*" to match your environment
mkfs.xfs /dev/nvme6n1
mount /dev/nvme6n1 /sp/tsminst1

# Cloud Cache
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme1n1
pvcreate /dev/nvme10n1
vgcreate sp_cc /dev/nvme1n1 /dev/nvme10n1
lvcreate --stripes 2 --stripesize 16 --extents 100%FREE --name sp_cc_vg sp_cc
mkfs.xfs /dev/mapper/sp_cc-sp_cc_vg
mount /dev/mapper/sp_cc-sp_cc_vg /sp/sp_cc

# DB
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme3n1
pvcreate /dev/nvme4n1
pvcreate /dev/nvme8n1
pvcreate /dev/nvme11n1
vgcreate sp_db /dev/nvme3n1 /dev/nvme4n1 /dev/nvme8n1 /dev/nvme11n1
lvcreate --stripes 4 --stripesize 16 --extents 170239 --name sp_db_db1 sp_db
```

```
lvcreate --stripes 4 --stripesize 16 --extents 170239 --name sp_db_db2 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 170239 --name sp_db_db3 sp_db
lvcreate --stripes 4 --stripesize 16 --extents 170236 --name sp_db_db4 sp_db
mkfs.ext4 /dev/mapper/sp_db-sp_db_db1
mkfs.ext4 /dev/mapper/sp_db-sp_db_db2
mkfs.ext4 /dev/mapper/sp_db-sp_db_db3
mkfs.ext4 /dev/mapper/sp_db-sp_db_db4
mount /dev/mapper/sp_db-sp_db_db1 /sp/sp_db1
mount /dev/mapper/sp_db-sp_db_db2 /sp/sp_db2
mount /dev/mapper/sp_db-sp_db_db3 /sp/sp_db3
mount /dev/mapper/sp_db-sp_db_db4 /sp/sp_db4

# Active Log
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme9n1
mount /dev/nvme9n1 /sp/sp_alog

# DB backup and Archive log
# Change the value for "nvme*" to match your environment
mkfs.ext4 /dev/nvme2n1
mkfs.ext4 /dev/nvme5n1
mkfs.ext4 /dev/nvme7n1
mount /dev/nvme2n1 /sp/sp_dbb1
mount /dev/nvme5n1 /sp/sp_dbb2
mount /dev/nvme7n1 /sp/sp_archlog

# Edit /etc/fstab to ensure that mount points will be restored on reboot
vi /etc/fstab

##############################################
#   Spectrum Protect Instance User Setup
##############################################

# Run the following as the "root" user.
# Add a Linux group and user to own the IBM Spectrum Protect instance.
# Set the password for this instance.
groupadd tsmsrvrs
useradd -g tsmsrvrs tsminst1
```

```
passwd tsminst1

# Ensure that this user can use the mount points previously created
chown -R tsminst1:tsmsrvrs /sp/

# From here, proceed with the IBM Spectrum Protect installation and setup
```

IBM Spectrum Protect, Extra-Small Amazon EC2 System, Ubuntu Linux: m5a.xlarge Instance Type

```
##############################################
#   Miscellaneous
##############################################

# Update apt packages, install tooling
apt update
apt upgrade -y
apt install scsitools -y
apt install dstat -y
apt install sysstat -y
apt install multipath-tools -y
apt install multipath-tools-boot -y

# Start the multipath daemon
touch /etc/multipath.conf
systemctl restart multipath-tools.service
# Set server hostname
hostnamectl set-hostname <desired hostname>

# Set vm.swappiness to 5 to match v4.1 xLinux blueprint recommendation
sysctl vm.swappiness=5

# Install the Korn shell
apt install ksh -y
# Reboot the system

##############################################
#   Spectrum Protect Disk Setup (Ubuntu)
##############################################
```

```
# Run the following as the "root" user

# Install LVM for Ubuntu
apt install lvm2 -y
systemctl enable lvm2-lvmetad
systemctl start lvm2-lvmetad
# Make the required Spectrum Protect directories
mkdir /sp
mkdir /sp/tsminst1
mkdir /sp/sp_db1
mkdir /sp/sp_db2
mkdir /sp/sp_db3
mkdir /sp/sp_db4
mkdir /sp/sp_dbb1
mkdir /sp/sp_alog
mkdir /sp/sp_archlog
mkdir /sp/sp_cc

# Note that disk name values may be different from those seen here in actual
deployments

# Instance directory
# Change the value for "nvme*" to match your environment
mkfs.xfs /dev/nvme6n1
mount /dev/nvme6n1 /sp/tsminst1

# Cloud Cache
pvcreate /dev/nvme2n1
vgcreate sp_cc /dev/nvme2n1
lvcreate --stripesize 16 --extents 100%FREE --name sp_cc_vg sp_cc
mkfs.xfs /dev/mapper/sp_cc-sp_cc_vg
mount /dev/mapper/sp_cc-sp_cc_vg /sp/sp_cc

# DB and actlog.
# Change the value for "nvme*" to match your environment
pvcreate /dev/nvme3n1
pvcreate /dev/nvme6n1
pvcreate /dev/nvme7n1
```

```
pvcreate /dev/nvme8n1
pvcreate /dev/nvme9n1
vgcreate sp_db /dev/nvme3n1 /dev/nvme6n1 /dev/nvme7n1 /dev/nvme8n1 /dev/nvme9n1
lvcreate --stripes 5 --stripesize 16 --extents 16385 --name sp_db_db1 sp_db
lvcreate --stripes 5 --stripesize 16 --extents 16385 --name sp_db_db2 sp_db
lvcreate --stripes 5 --stripesize 16 --extents 16385 --name sp_db_db3 sp_db
lvcreate --stripes 5 --stripesize 16 --extents 16385 --name sp_db_db4 sp_db
lvcreate --stripes 5 --stripesize 16 --extents 16375 --name sp_db_alog sp_db
mkfs.ext4 /dev/mapper/sp_db-sp_db_db1
mkfs.ext4 /dev/mapper/sp_db-sp_db_db2
mkfs.ext4 /dev/mapper/sp_db-sp_db_db3
mkfs.ext4 /dev/mapper/sp_db-sp_db_db4
mkfs.ext4 /dev/mapper/sp_db-sp_db_alog
mount /dev/mapper/sp_db-sp_db_db1 /sp/sp_db1
mount /dev/mapper/sp_db-sp_db_db2 /sp/sp_db2
mount /dev/mapper/sp_db-sp_db_db3 /sp/sp_db3
mount /dev/mapper/sp_db-sp_db_db4 /sp/sp_db4
mount /dev/mapper/sp_db-sp_db_alog /sp/sp_alog

# DB backup and Archive log
pvcreate /dev/nvme3n1
vgcreate sp_dbb_vg /dev/nvme3n1
lvcreate --extents 50%FREE --name sp_dbb_dbb sp_dbb_vg
lvcreate --extents 100%FREE --name sp_dbb_archlog sp_dbb_vg
mkfs.ext4 /dev/mapper/sp_dbb_vg-sp_dbb_dbb
mkfs.ext4 /dev/mapper/sp_dbb_vg-sp_dbb_archlog
mount /dev/mapper/sp_dbb_vg-sp_dbb_dbb /sp/sp_dbb1
mount /dev/mapper/sp_dbb_vg-sp_dbb_archlog /sp/sp_archlog

# Edit /etc/fstab to ensure that mount points will be restored on reboot
vi /etc/fstab

##############################################
#   Spectrum Protect Instance User Setup
##############################################

# Run the following as the "root" user.
# Add a Linux group and user to own the IBM Spectrum Protect instance.
```

```
# Set the password for this instance.
groupadd tsmsrvrs
useradd -g tsmsrvrs tsminst1
passwd tsminst1

# Ensure that this user can use the mount points previously created
chown -R tsminst1:tsmsrvrs /sp/

# From here, proceed with the IBM Spectrum Protect installation and setup
```

Disk Benchmarking

As a part of vetting each AWS test configuration outlined in this document, disk benchmark tests were performed to validate the capability of the disk volumes underlying the IBM Spectrum Protect database and cloud accelerator cache. From a database point of view, this vetting was done to ensure that the volumes were sufficiently capable from an IOPS perspective to support the 8 KiByte random mixed write and read workload that a busy Blueprint-level system would demand. From a cloud cache standpoint, the vetting was performed to ensure that overlapped 128-256 KiByte write and read throughput could achieve a rate high enough such that the server's bottleneck for IO would be at the instance-to-object storage network level and not the disk level. The goal was to ensure that the disk could perform at a rate such that the IBM Spectrum Protect server could utilize it during overlapped ingest and be able to stress the network link layer simultaneously.

Disk benchmarking was performed by using the tsmdiskperf.pl Perl script, provided as a part of the Blueprint configuration scripts package found on the IBM Spectrum Protect Blueprints page (References [1]). Execution of the script was performed as follows:

```
perl tsmdiskperf.pl workload=stgpool fslist=directory_list
perl tsmdiskperf.pl workload=db fslist=directory_list
```

With a `stgpool` workload specification, the script drives a 256 KiByte IO pattern, whereas with a `db` workload specification, the script drives 8 KiByte operations. For each directory location provided as a value to the comma-separate `fslist`, a pair of IO processes is created to perform writes and reads to test files that are generated in that directory.

Typical script output for a `stgpool` workload run resembles the following example:

```
================================================================
: IBM Spectrum Protect disk performance test    (Program version 3.1b)
:
: Workload type:                stgpool
: Number of filesystems:        1
```

```
: Mode:                          readwrite
: Files to write per fs:         5
: File size:                     2 GB
:
================================================================================
:
: Beginning I/O test.
: The test can take upwards of ten minutes, please be patient ...
: Starting write thread ID: 1 on filesystem /sp/sp_cc/1
: Starting read thread ID: 2 on filesystem /sp/sp_cc/1
: All threads are finished.  Stopping iostat process with id 111519
====================================================================
:  RESULTS:
:  Devices reported on from  output:
:  dm-2
:
:  Average R Throughput (KB/sec):       19473.92
:  Average W Throughput (KB/sec):       19377.45
:  Avg Combined Throughput (MB/sec):    37.94
:  Max Combined Throughput (MB/sec):    160.57
:
:  Average IOPS:                        464.63
:  Peak IOPS:                           2154.57 at 11/10/2017 04:22:32
:
:  Total elapsed time (seconds):        443
====================================================================
```

The value that was extracted for the purposes of comparison and validation for `stgpool` workloads was `Avg Combined Throughput (MB/sec)`. The goal was to determine the largest aggregate average throughput for writes and reads to the accelerator cache disk such that overlapped backup ingest and transfer to object storage will not be constrained by disk capability.

When running the tool in `db` workload mode, output should appear similar to the following example:

```
================================================================================
: IBM Spectrum Protect disk performance test    (Program version 3.1b)
:
```

```
: Workload type:                db

: Number of filesystems:        1

: Mode:                         readwrite

: Thread count per FS:          2

: I/Os per thread:              500000

: File size:                    10 GB

:

=====================================================================

:

: Creating files of 10 GB to simulate IBM Spectrum Protect DB.

: Issuing command ./ldeedee if=/dev/zero of=/sp/sp_db1/1/tsmdiskperf_1 bs=1024k
count=10240 dio=2 > /dev/null 2>&1

: Issuing command ./ldeedee if=/dev/zero of=/sp/sp_db1/1/tsmdiskperf_2 bs=1024k
count=10240 dio=2 > /dev/null 2>&1

:

: Beginning I/O test.

: The test can take upwards of ten minutes, please be patient ...

: All threads are finished.  Stopping iostat process with id 111978

=====================================================================

:  RESULTS:

:  Devices reported on from  output:

:  dm-6

:

:  Average R Throughput (KB/sec):       12907.53

:  Average W Throughput (KB/sec):       12707.15

:  Avg Combined Throughput (MB/sec):    25.01

:  Max Combined Throughput (MB/sec):    42.70

:

:  Average IOPS:                        3202.28

:  Peak IOPS:                           5465.86 at 11/10/2017 04:31:47

:

:  Total elapsed time (seconds):        30

=====================================================================
```

For the db workload tests, the `Avg Combined Throughput (MB/sec)` and `Average IOPS` metrics are significant for evaluating database disk capability. Here, the small random IOPS capability of the underlying disk that is used for the IBM Spectrum Protect Db2 database is of interest.

To conduct measurements of your own, increase the number of write/read threads pairs (and directories) by 1 for each test until the average throughput, the average IOPS, or both stabilize (level off). Benchmark test results are provided here as a reference for those who want to build systems resembling those laid out in this document and who want to validate that their system is capable of supporting the described level of ingestion. For each graph, the horizontal axis represents the quantity of write/read thread pairs (and the number of directory locations used with `fslist`). For each successive bar to the right, the thread count affecting the disk is increased by 2 (1 write thread, 1 read thread, and adding a directory location). The vertical axis represents total average throughput in MiBytes/s.

AWS Large Configuration

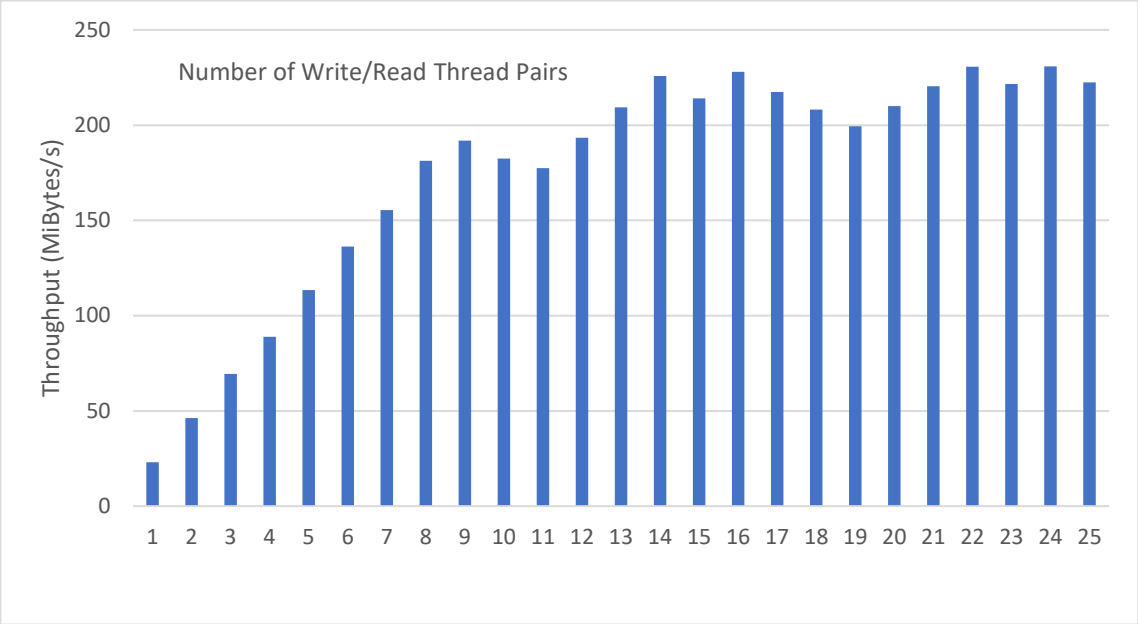


Figure 5: AWS large configuration; database volume average throughput; 8 KiByte random writes/reads

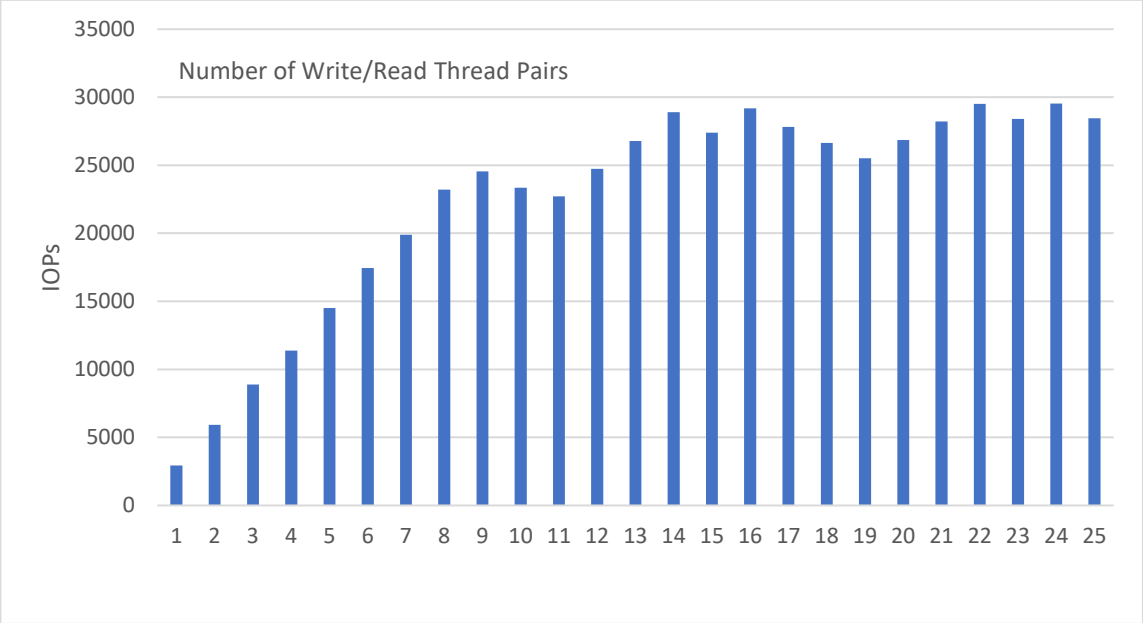


Figure 6: AWS large configuration; database volume average IOPS; 8 KiByte random writes/reads

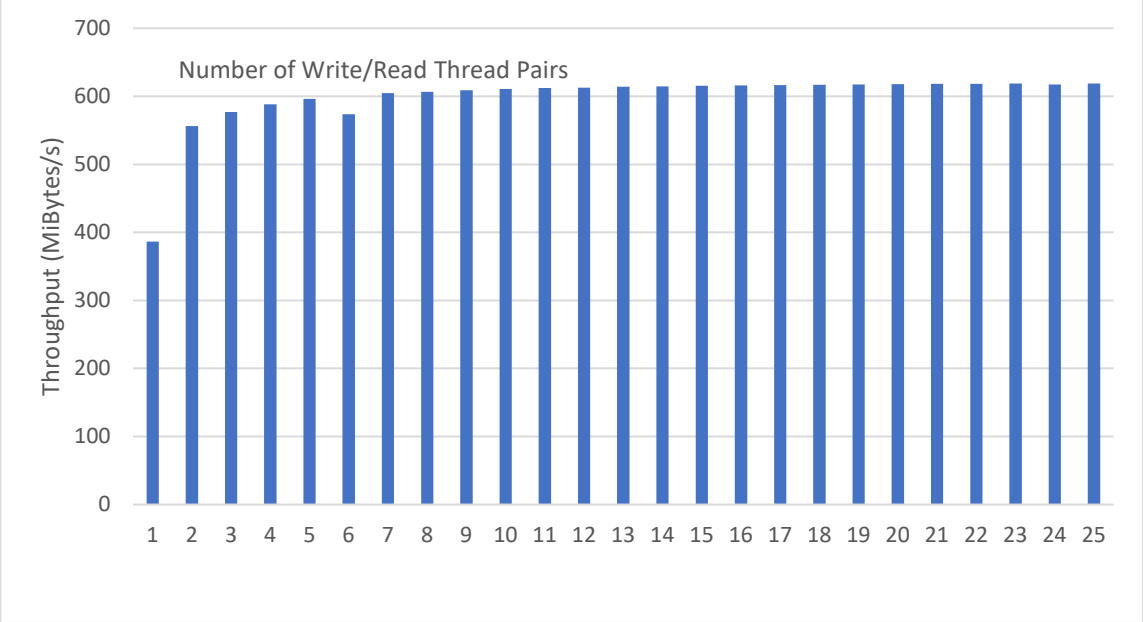


Figure 7: AWS large configuration; cloud cache volume average throughput; mixed 256 KiByte writes and reads

AWS Medium Configuration

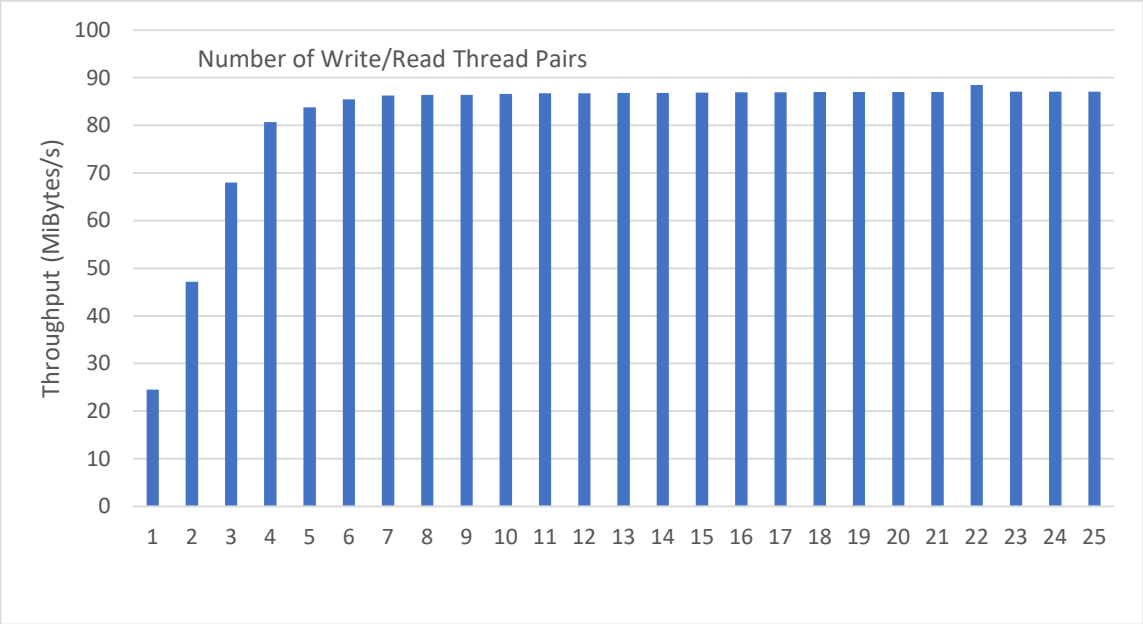


Figure 8: AWS medium configuration; database volume average throughput; 8 KiByte random writes/reads

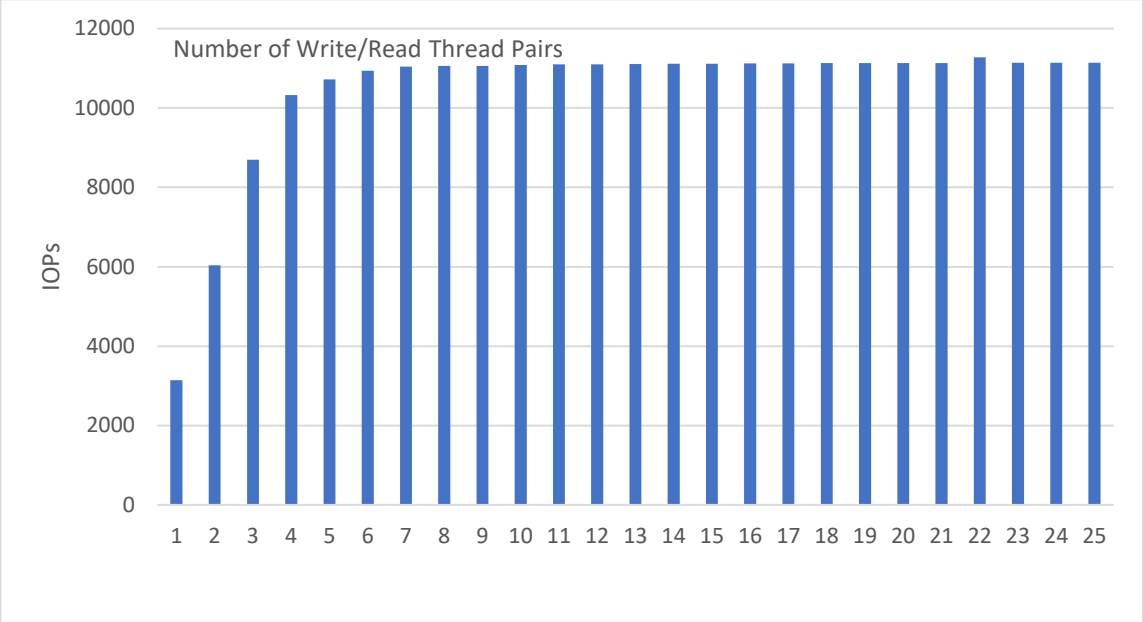


Figure 9: AWS medium configuration; database volume average IOPS; 8 KiByte random writes/reads

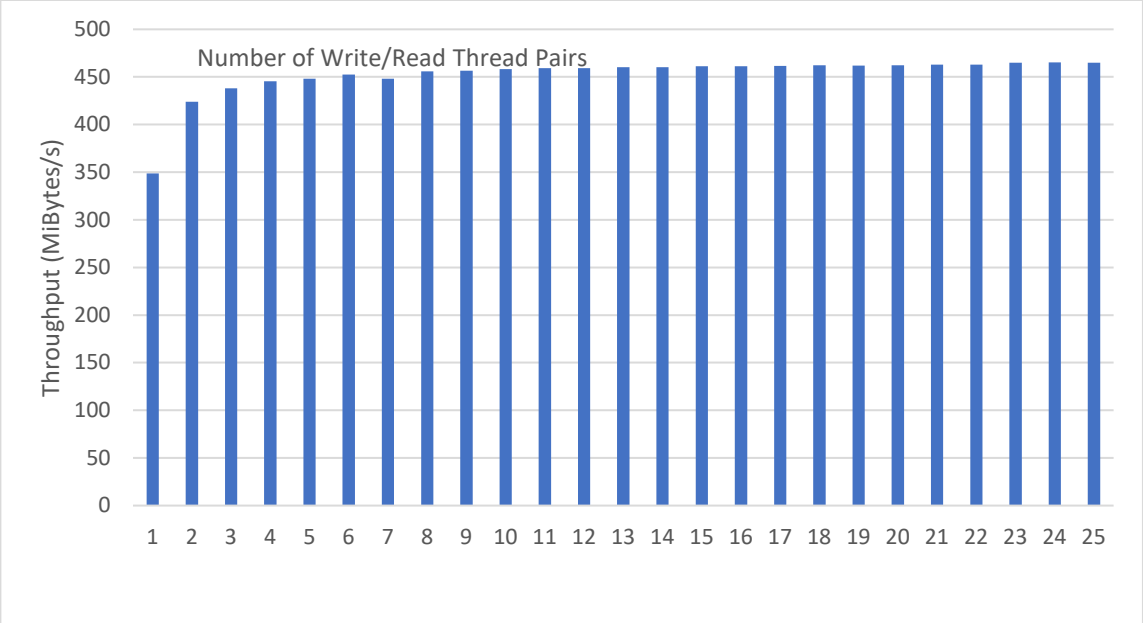


Figure 10: AWS medium configuration; cloud cache volume average throughput; mixed 256 KiByte writes and reads

AWS Small Configuration

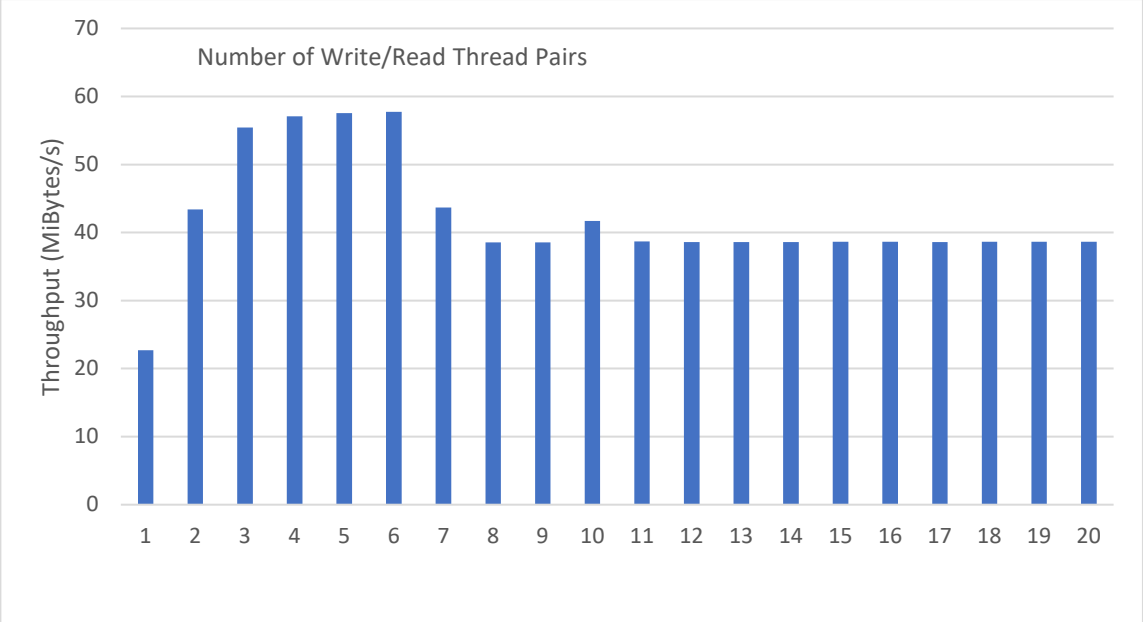


Figure 11: AWS small configuration; database volume average throughput; 8 KiByte random writes/reads

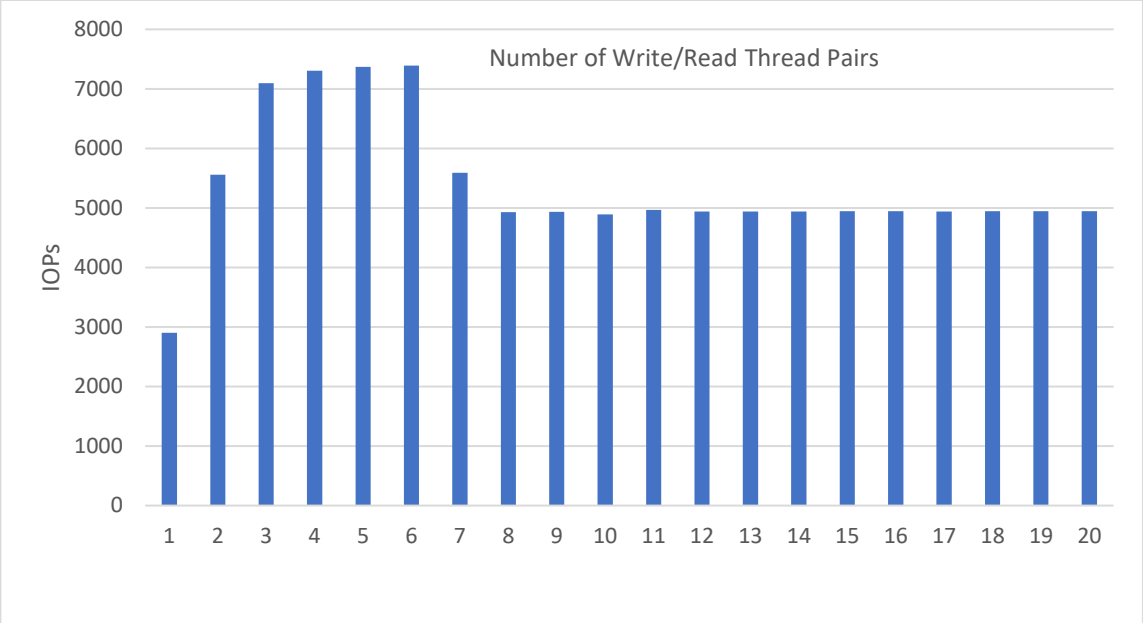


Figure 12: AWS small configuration; database volume average IOPS; 8 KiByte random writes/reads

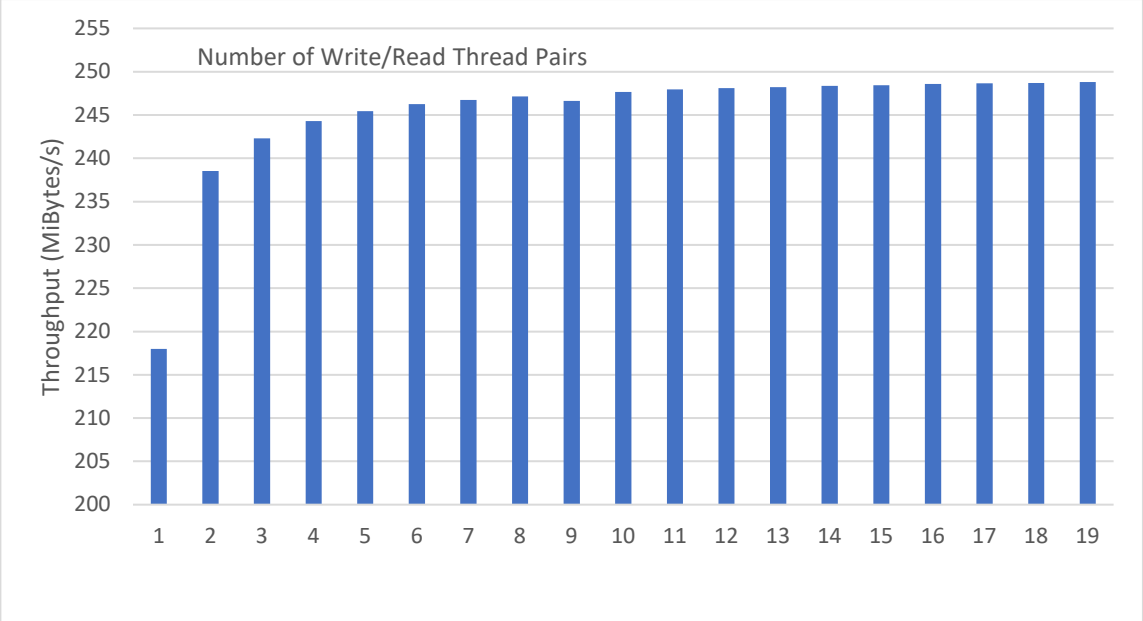


Figure 13: AWS small configuration; cloud cache volume average throughput; mixed 256 KiByte writes/reads

Object Storage Benchmarking

Another important step in validating the capability of an IBM Spectrum Protect in-the-cloud solution is to benchmark the throughput of the bare metal server or compute instance to the chosen target object storage system with a workload that is typical of IBM Spectrum Protect. Ideally, any in-the-cloud IBM Spectrum Protect solution should be network bound

in terms of its connection to object storage. Post inline data deduplication, compression, and encryption, the back-end ingestion rate over HTTPS should dictate an upper bound for daily ingestion performance of a system.

To help facilitate this test activity in-house, a Java program was developed by the IBM Spectrum Protect test team to emulate the behavior of the server's use of the Amazon S3 Java API. The tool can be used to drive various backup and restore-type activities to object storage, including direct HTTP PUT, GET, multipart file upload, and range-read restore behavior with a variable number of threads. This tool, known as `SPObjBench.jar`, is included with the Benchmarking package provided with this document.

Also included in the Benchmarking package is a Perl script, `tsmobjperf.pl`, which can be used to automate execution of the `SPObjBench.jar` file with several thread counts to measure ingest (PUT) and restore (GET) scalability.

On the normal ingestion path within the scope of a direct-to-cloud with accelerator cache architecture, the IBM Spectrum Protect server attempts to upload up to 100 1 GB disk container files from the accelerator cache in parallel by using multipart upload. Within a production environment, this work would occur in conjunction with overlapped ingestion to another set of container files within the same storage pool directory locations on accelerator cache disk storage.

To attempt to isolate and gauge the instance-to-object storage ingestion capability of a system, you can complete the following steps:

1. Populate a set of 10 1 GB files in a memory-mapped file system location to use as source data for ingestion. The use of memory-mapped locations (such as `tmpfs` on Linux) is preferred to eliminate source disk bottlenecks. For a Linux system with at least 11 GB of free RAM, run the following commands:

   ```
   mkdir /mnt/ramdisk

   mount –t tmpfs –o size=11g tmpfs /mnt/ramdisk

   for I in `seq 10`; do dd if=/dev/urandom
   of=/mnt/ramdisk/file.$I bs=1048576 count=1024; done
   ```

2. To run a set of automated tests scaling from 1 to 100 threads, run the `tsmobjperf.pl` tool by using the recently created RAM disk files as source files to upload. If more threads are specified than files are present in the source list, the tool completes a round-robin action over these source files. Because all activity is read-only, using separate file handles from memory-mapped sources, multiple threads sharing the same file is not a concern. To test with 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 threads, run the tool as follows, specifying the arguments as needed:

   `perl tsmobjperf.pl type=`*`type`* `endpoints=`*`endpoint`* `user="`*`user`*`" pass="`*`pass`*`" bucket=`*`bucket`* *`min`*`=1 `*`max`*`=100 `*`step`*`=10 flist=`*`comma_delimited_source_files_list`*

   where:

   - *type* should be `s3` for Amazon S3.

- *endpoint* specifies a comma-delimited list of IP addresses or URLs for the object storage endpoints. For Amazon S3, this should be a single private S3 endpoint that is accessed over HTTPS (for security), preferably with a VPC endpoint set up.
- For S3, the *user* is a public key ID.
- For S3, the *pass* is the secret key for a user who has valid S3 credentials to create buckets and PUT and GET objects in the region indicated by the *endpoint* URL. These values align with those that are used to define an IBM Spectrum Protect cloud-container storage pool, either via the Operations Center or the command line.
- The *bucket* value should be an Amazon S3 bucket name that the credentialed user has create/PUT/GET access to and that exists in the object storage system. The *min* and *max* values should indicate the minimum and maximum thread counts to test.
- The *step* value should indicate the increase in thread count from test to test.
- The `flist` parameter should include a comma-delimited list of source files to be used for multipart upload. These files should be the same as those created earlier in the memory-mapped file system.

The following example is for execution of an Amazon S3 based endpoint in the US West (Oregon) Region, using 100 upload threads with an existing test bucket:

```
perl tsmobjperf.pl type=s3 endpoints=https://s3.us-west-
2.amazonaws.com user=”PUBLICKEYID” pass=”SECRETKEY”
bucket=testbucket min=1 max=100 step=10
flist=/mnt/ramdisk/file.1,/mnt/ramdisk/file.2,/mnt/ramdisk/file.
3,/mnt/ramdisk/file.4,/mnt/ramdisk/file.5,/mnt/ramdisk/file.6,/m
nt/ramdisk/file.7,/mnt/ramdisk/file.8,/mnt/ramdisk/file.9
,/mnt/ramdisk/file.10
```

Each thread count test (for 1, 10, 20, or more threads) uploads 10 x 1 GB objects per thread. The previous example would result in a total of 5510 GB of data being stored to the test bucket after all thread tests are completed. The tool does not remove objects that are created. You must remove the objects manually after test completion.

Upon completion, the tool generates aggregate throughput metrics that can be used to estimate practical instance-to-object storage performance rates for IBM Spectrum Protect. Data is provided in comma-separated-value format (CSV) and the output of the `SPObjBench.jar` tool can be inspected upon completion as well:

```
================================================================
: IBM Spectrum Protect object storage test
:
: Test Mode:    write
: Type:         s3
: Endpoints:    https:// s3.us-west-2.amazonaws.com
```

```
: User:         PUBLICKEY
: Pass:         SECRETKEY
: Test Bucket:  testbucket
: Min Threads:  1
: Max Threads:  100
: Thread Step:  10
: File List:
/mnt/ramdisk/file.1,/mnt/ramdisk/file.2,/mnt/ramdisk/file.3,/mnt/ramdisk/file.4,
/mnt/ramdisk/file.5,/mnt/ramdisk/file.6,/mnt/ramdisk/file.7,/mnt/ramdisk/file.8,
/mnt/ramdisk/file.9 ,/mnt/ramdisk/file.10
: Using java:   java
======================================================================

SPObjBench.jar output being captured to file:   tsmobjperf.1540336631.out

======================================================================
: Test Results
Thread Count, Write Throughput (MB/s)
1, XXX
10, XXX
20, XXX
30, XXX
40, XXX
50, XXX
60, XXX
70, XXX
80, XXX
90, XXX
100, XXX
======================================================================
```

It can be beneficial to monitor network transmission rates externally from the tool, as well, to validate the absolute throughput rate that is experienced to object storage over the (Ethernet) network. The tool reports an aggregate rate that can include build-up and tear-down overhead associated with the tool. Calculating an actual transmission rate from the instance-to-object storage while the test is running can give an indication of the throughput limits of the environment. On Linux, for example, the `dstat` utility can be used to monitor several system metrics at once, including network interface send and receive statistics, by using the basic command:

```
% dstat
You did not select any stats, using -cdngy by default.
----total-cpu-usage---- -dsk/total- -net/total- ---paging-- ---system--
usr sys idl wai hiq siq| read  writ| recv  send|  in   out | int   csw
  0   0 100   0   0   0|  60B 2511B|   0     0 |   0     0 |  76    71
 15   1  84   0   0   1|   0    24k|1674k   58M|   0     0 |  42k 2785
 15   1  83   0   0   1|   0     0 |1838k   62M|   0     0 |  46k 2969
 16   1  82   0   0   1|   0     0 |1832k   61M|   0     0 |  45k 3127
 15   1  84   0   0   1|   0     0 |1753k   61M|   0     0 |  44k 2822
 16   1  83   0   0   1|   0     0 |1811k   62M|   0     0 |  45k 3001
 15   1  83   0   0   1|   0     0 |1778k   62M|   0     0 |  45k 3068
 16   1  82   0   0   1|   0     0 |1870k   63M|   0     0 |  46k 3068
 16   1  82   0   0   1|   0     0 |1933k   64M|   0     0 |  46k 3011
 15   1  83   0   0   1|   0     0 |1834k   63M|   0     0 |  46k 2974
```

The `dstat` tool outputs a new line of metrics at a configured interval, much like the standard `iostat` and `netstat` utilities. For the execution above, the `net/total send` column is of greatest interest, here reported in MiBytes, as an indication of how quickly data could be sent to the object storage endpoint from the server.

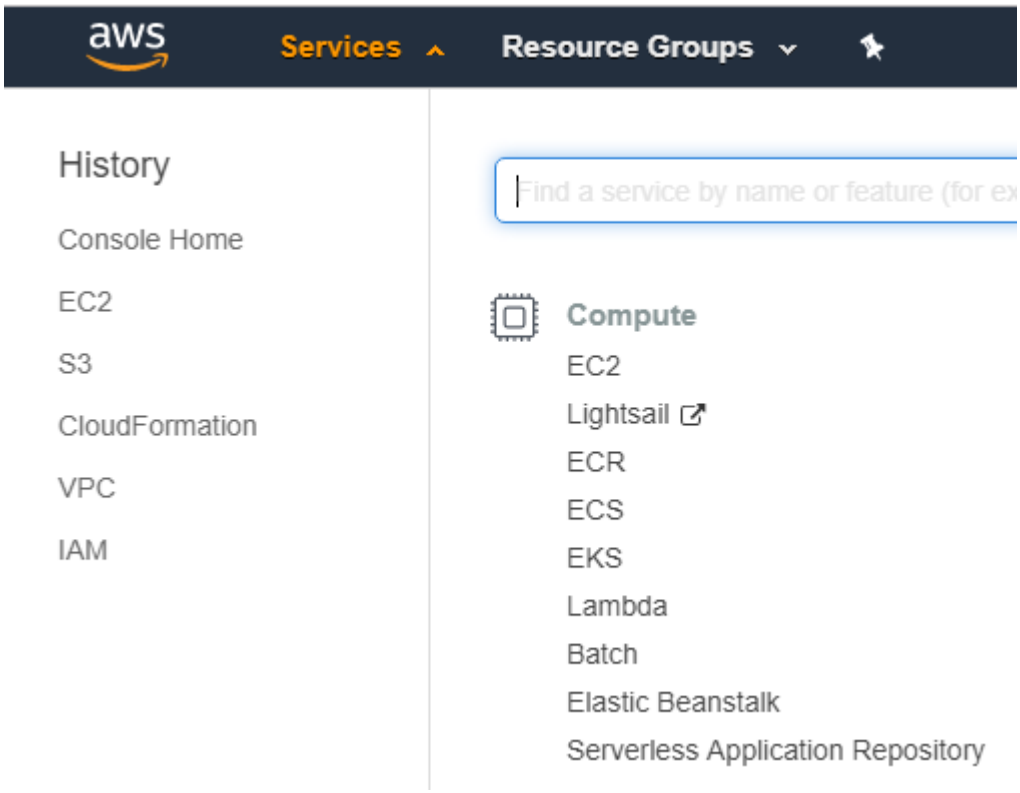
Instance and Object Storage: Navigating the AWS Portal

When deploying a new IBM Spectrum Protect environment based on AWS, you must navigate the web portal and create relevant cloud resources. Use the following steps as a starting point for creating AWS resources. An important consideration when building an IBM Spectrum Protect server in AWS is to correctly configure host servers and object storage such that the private (internal AWS) network link between these resources is efficient.

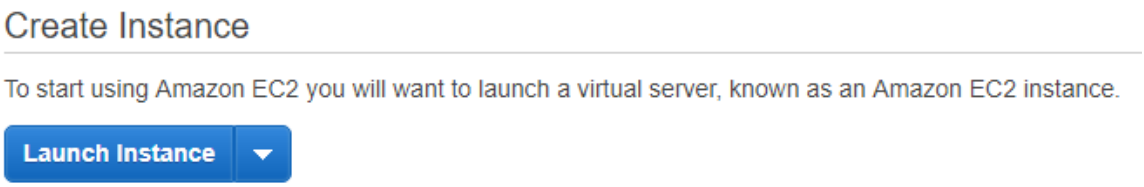
Begin by navigating to the AWS home page and sign in with the appropriate user credentials.

AWS EC2 Instances

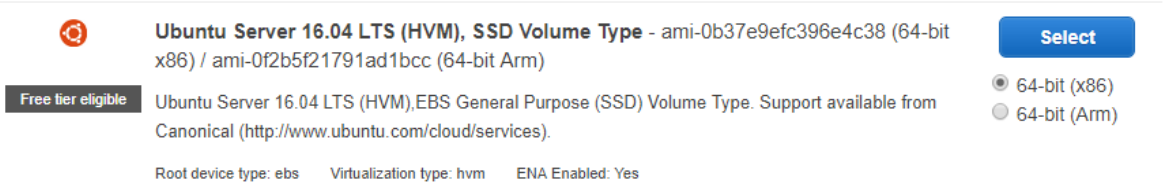
1. At the top left, click **Services**. In the **Compute** section, click **EC2**.



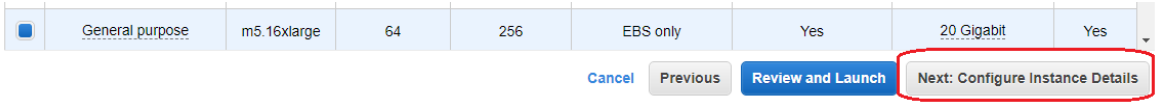
2. In the **Create Instance** section, click **Launch Instance**.



3. Select an appropriate and supported Amazon Machine Image (AMI), for example, Ubuntu Server 16.04 LTS.

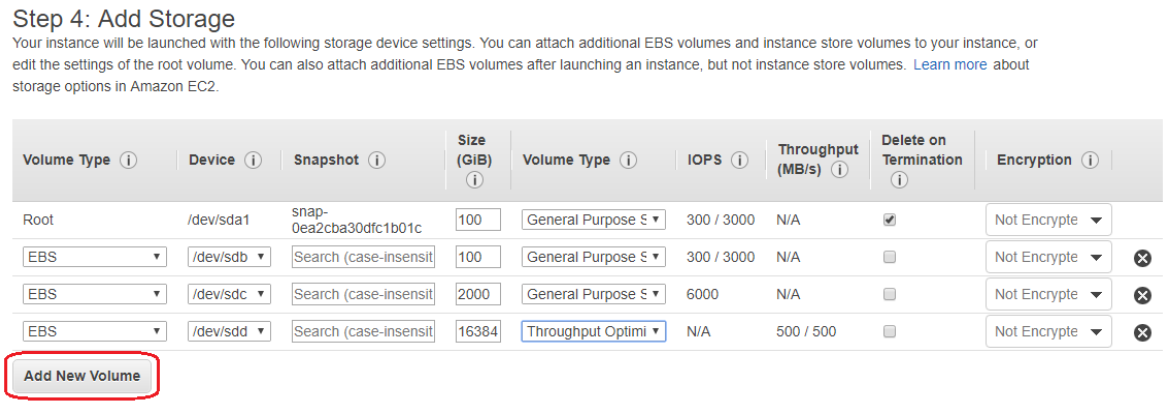


4. Select an instance type that is applicable to the desired IBM Spectrum Protect cloud Blueprint size. Click **Next: Configure Instance Details**.



5. On the “Step 3: Configure Instance Details” page, customize the VPC, subnet, IAM role, and other settings related to the instance. Click **Next: Add Storage**.

6. On the “Step 4: Add Storage” page, add the necessary IBM Spectrum Protect EBS block disk appropriate to the desired cloud Blueprint size. Optionally, select the **delete on termination** check box if EBS volumes should be deleted when the instance is deleted.
   Warning: Deletion of EBS volumes might not be desirable with IBM Spectrum Protect. EBS volumes contain persistent IBM Spectrum Protect storage pool and database data that might be required outside the lifetime of the instance. For example, it might be necessary to terminate an EC2 instance and provision one of a different size or model in some circumstances. Maintaining EBS disks with IBM Spectrum Protect data separate from the instance will allow for greater instance flexibility.



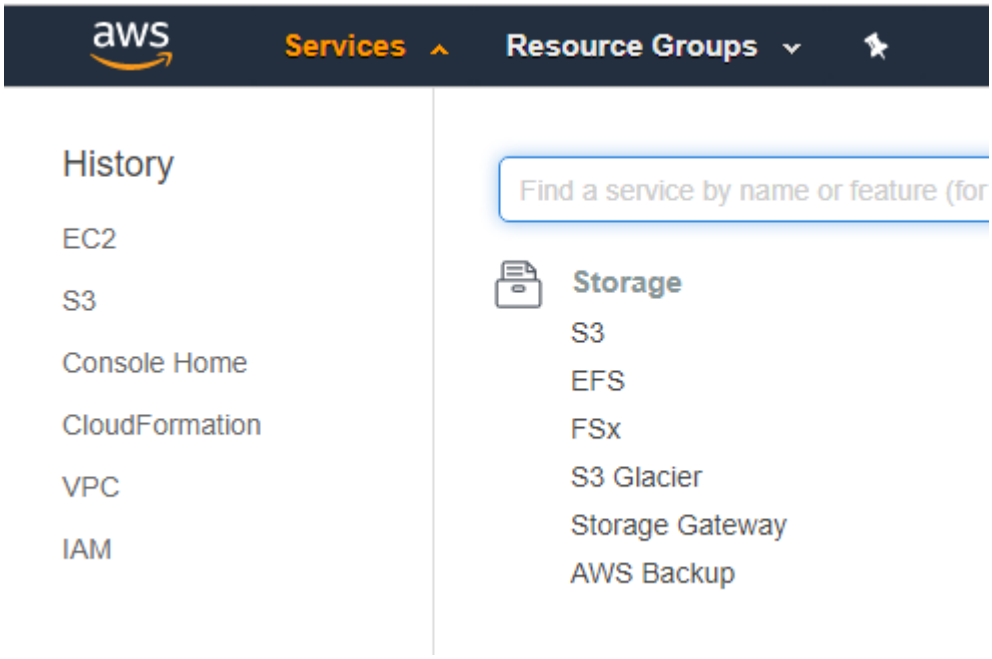
7. Click **Next: Add Tags**. Add any instance-specific tags that you require. Then, click **Next: Configure Security Group**.

8. On the “Step 6: Configure Security Group” page, assign the instance to a new or existing Amazon EC2 security group. Security groups help to organize incoming or outgoing protocol and port restriction rules for one or more instances in a group. Note that outgoing TCP port 80 is required for HTTP communication to S3 object storage and TCP port 443 is required for HTTPS. Click **Review and Launch**.

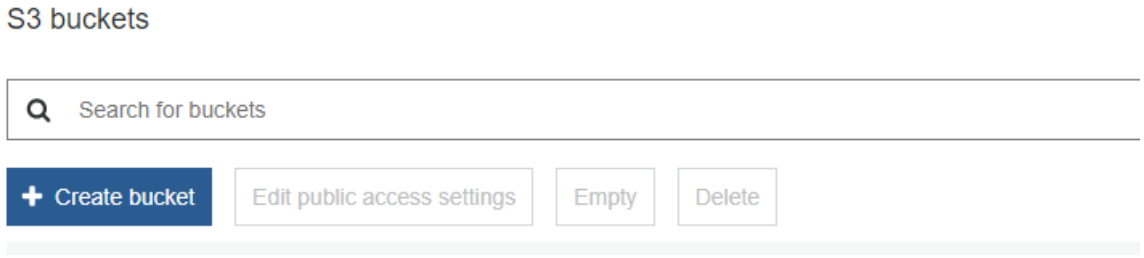
9. On the final “Step 7: Review Instance Launch” page, review the desired instance settings. If the settings are correct, click **Launch** to direct Amazon to create the instance and attached volumes.

AWS S3 Object Storage

1. At the top left, click **Services**. Then, in the **Storage** section, click **S3**.

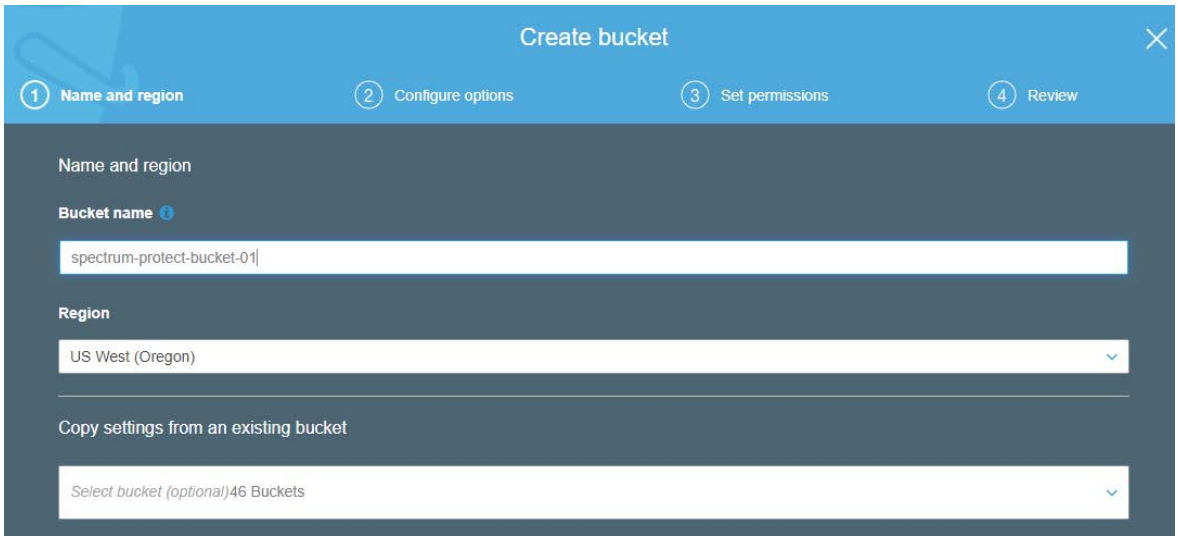


2. Click **Create bucket** to create a bucket for use by IBM Spectrum Protect cloud-container storage pools.



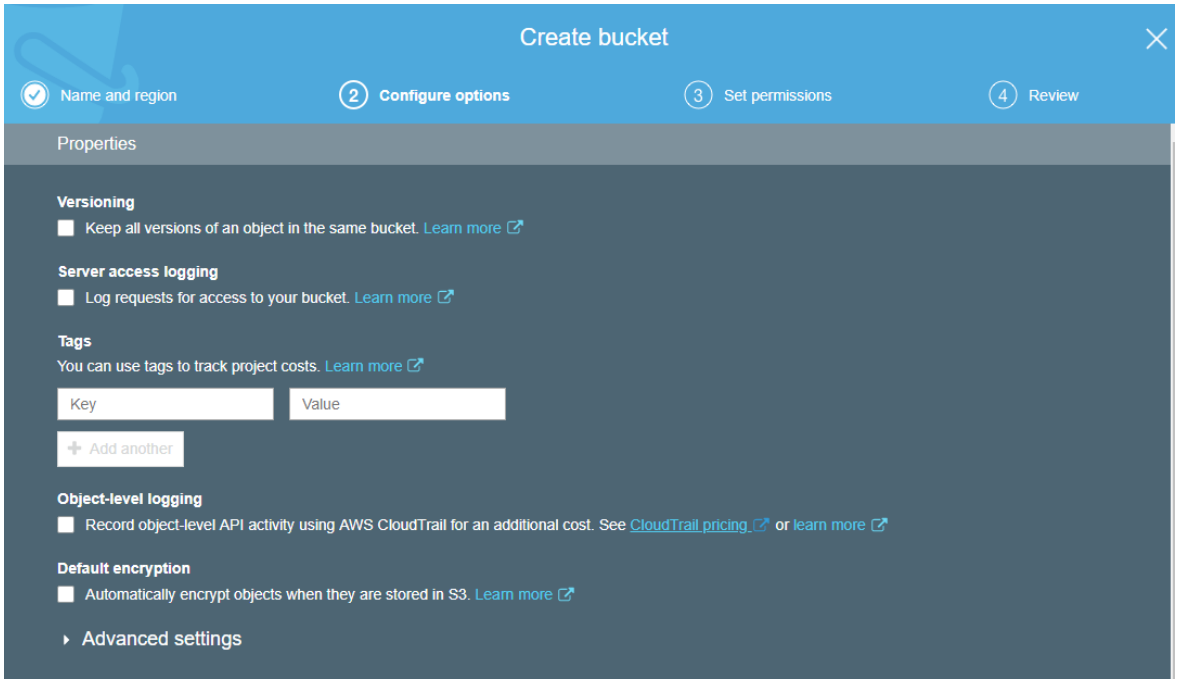
3. Specify a globally-unique name for the bucket and select an Amazon Region where the bucket will be created. For optimal performance, the Amazon S3 bucket and Amazon EC2 instance hosting IBM Spectrum Protect should be in the same Amazon

Region. Click **Next**.



4. On the “Configure options” pane of the create bucket wizard, specify any further options that you require.
   Tips:
   - Versioning is not recommended with IBM Spectrum Protect because this feature is not used by cloud-container storage pools.
   - If data privacy is a concern, object-level encryption is not necessary because deduplicated extent-level encryption can be enabled on the IBM Spectrum Protect cloud-container storage pool with little, if any, performance penalty.

   Click **Next**.



5. On the “Set Permissions” page, adjust public bucket access, if desired. The preferred method is to keep the default settings. Click **Next**.

6. On the final “Review” page, review the desired bucket settings. If the settings are correct, click **Create bucket** to direct Amazon to create the bucket within the desired region.

When you access Amazon S3 object storage from an Amazon EC2 instance, be sure to use the optimal S3 protocol endpoint. For a listing of available endpoints, see References [8]. If possible, use the private endpoint that is geographically closest to the Amazon EC2 compute instance to help optimize throughput.

REFERENCES

[1] IBM Spectrum Protect Blueprints:
https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Tivoli%20Storage%20Manager/page/IBM%20Spectrum%20Protect%20Blueprints

[2] Amazon Web Services:
https://aws.amazon.com

[3] Amazon EC2, "Resizing an Amazon EBS-backed Instance":
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-resize.html

[4] Amazon EBS-Optimized Instances:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html

[5] Amazon EBS Volume Types:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html

[6] AWS Direct Connect:
https://aws.amazon.com/directconnect/

[7] Overview - IBM Spectrum Protect Supported Operating Systems:
http://www.ibm.com/support/docview.wss?uid=swg21243309

[8] AWS Service Endpoints:
https://docs.aws.amazon.com/general/latest/gr/rande.html

[9] Amazon Nitro System:
https://aws.amazon.com/ec2/nitro/

[10] Announcing S3 Intelligent-Tiering:
https://aws.amazon.com/about-aws/whats-new/2018/11/s3-intelligent-tiering/

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

Trademarks