Tool and application guides

StorageGRID solutions and resources

NetApp
December 10, 2025

Table of Contents

Tool and application guides

Use Cloudera Hadoop S3A connector with StorageGRID

By Angela Cheng

Hadoop has been a favorite of data scientists for some time now. Hadoop allows for the distributed processing of large data sets across clusters of computers using simple programing frameworks. Hadoop was designed to scale up from single servers to thousands of machines, with each machine possessing local compute and storage.

Why use S3A for Hadoop workflows?

As the volume of data has grown over time, the approach of adding new machines with their own compute and storage has become inefficient. Scaling linearly creates challenges for using resources efficiently and managing the infrastructure.

To address these challenges, the Hadoop S3A client offers high-performance I/O against S3 object storage. Implementing a Hadoop workflow with S3A helps you leverage object storage as a data repository and enables you to separate compute and storage, which in turn enables you to scale compute and storage independently. Decoupling compute and storage also enables you to dedicate the right amount of resources for your compute jobs and provide capacity based on the size of your data set. Therefore, you can reduce your overall TCO for Hadoop workflows.

Configure S3A connector to use StorageGRID

Prerequisites

- A StorageGRID S3 endpoint URL, a tenant s3 access key, and a secret key for Hadoop S3A connection testing.
- A Cloudera cluster and root or sudo permission to each host in the cluster to install the Java package.

As of April 2022, Java 11.0.14 with Cloudera 7.1.7 was tested against StorageGRID 11.5 and 11.6. However, the Java version number might be different at the time of a new install.

Install Java package

1. Check the Cloudera support matrix for the supported JDK version.
2. Download the Java 11.x package that matches the Cloudera cluster operating system. Copy this package to each host in the cluster. In this example, the rpm package is used for CentOS.
3. Log into each host as root or using an account with sudo permission. Perform the following steps on each host:
   a. Install the package:

      ```
      $ sudo rpm -Uvh jdk-11.0.14_linux-x64_bin.rpm
      ```

   b. Check where Java is installed. If multiple versions are installed, set the newly installed version as default:

```
alternatives --config java

There are 2 programs which provide 'java'.

  Selection    Command
-----------------------------------------------
 +1           /usr/java/jre1.8.0_291-amd64/bin/java
  2           /usr/java/jdk-11.0.14/bin/java

Enter to keep the current selection[+], or type selection number: 2
```

c. Add this line to the end of `/etc/profile`. The path should match the path of above selection:

```
export JAVA_HOME=/usr/java/jdk-11.0.14
```

d. Run the following command for the profile to take effect:

```
source /etc/profile
```

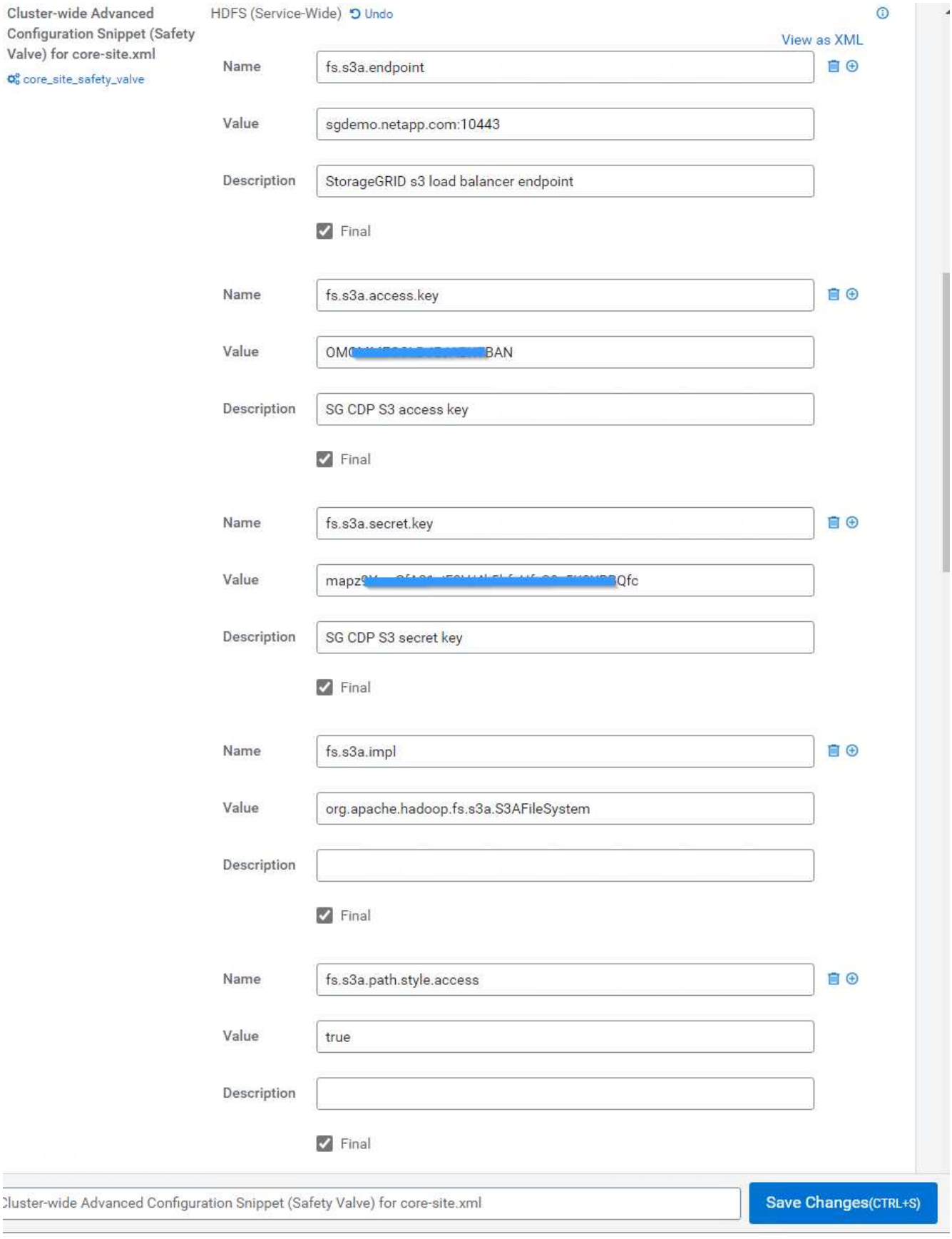
Cloudera HDFS S3A configuration

Steps

1. From the Cloudera Manager GUI, select Clusters > HDFS, and select Configuration.
2. Under CATEGORY, select Advanced, and scroll down to locate `Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml`.
3. Click the (+) sign and add following value pairs.

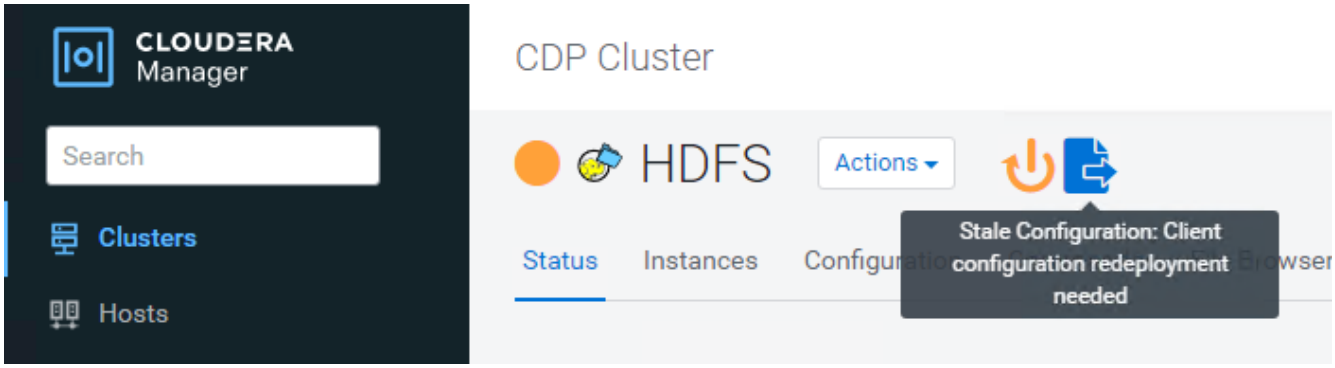
Name	Value
fs.s3a.access.key	*<tenant s3 access key from StorageGRID>*
fs.s3a.secret.key	*<tenant s3 secret key from StorageGRID>*
fs.s3a.connection.ssl.enabled	[true or false] (default is https if this entry is missing)
fs.s3a.endpoint	*<StorageGRID S3 endpoint:port>*
fs.s3a.impl	org.apache.hadoop.fs.s3a.S3AFileSystem
fs.s3a.path.style.access	[true or false] (default is virtual host style if this entry is missing)

Sample screenshot



4. Click the Save Changes button. Select the Stale Configuration icon from the HDFS menu bar, select

Restart Stale Services on the next page, and select Restart Now.



Test S3A connection to StorageGRID

Perform basic connection test

Log into one of the hosts in the Cloudera cluster, and enter `hadoop fs -ls s3a://<bucket-name>/`.

The following example uses path syle with a pre-existing hdfs-test bucket and a test object.

```
[root@ce-n1 ~]# hadoop fs -ls s3a://hdfs-test/
22/02/15 18:24:37 WARN impl.MetricsConfig: Cannot locate configuration:
tried hadoop-metrics2-s3a-file-system.properties,hadoop-
metrics2.properties
22/02/15 18:24:37 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot
period at 10 second(s).
22/02/15 18:24:37 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system started
22/02/15 18:24:37 INFO Configuration.deprecation: No unit for
fs.s3a.connection.request.timeout(0) assuming SECONDS
Found 1 items
-rw-rw-rw-   1 root root       1679 2022-02-14 16:03 s3a://hdfs-test/test
22/02/15 18:24:38 INFO impl.MetricsSystemImpl: Stopping s3a-file-system
metrics system...
22/02/15 18:24:38 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system stopped.
22/02/15 18:24:38 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system shutdown complete.
```

Troubleshooting

Scenario 1

Use an HTTPS connection to StorageGRID and get a `handshake_failure` error after a 15 minute timeout.

Reason: Old JRE/JDK version using outdated or unsupported TLS cipher suite for connection to StorageGRID.

Sample error message

```
[root@ce-n1 ~]# hadoop fs -ls s3a://hdfs-test/
22/02/15 18:52:34 WARN impl.MetricsConfig: Cannot locate configuration:
tried hadoop-metrics2-s3a-file-system.properties,hadoop-
metrics2.properties
22/02/15 18:52:34 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot
period at 10 second(s).
22/02/15 18:52:34 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system started
22/02/15 18:52:35 INFO Configuration.deprecation: No unit for
fs.s3a.connection.request.timeout(0) assuming SECONDS
22/02/15 19:04:51 INFO impl.MetricsSystemImpl: Stopping s3a-file-system
metrics system...
22/02/15 19:04:51 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system stopped.
22/02/15 19:04:51 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system shutdown complete.
22/02/15 19:04:51 WARN fs.FileSystem: Failed to initialize fileystem
s3a://hdfs-test/: org.apache.hadoop.fs.s3a.AWSClientIOException:
doesBucketExistV2 on hdfs: com.amazonaws.SdkClientException: Unable to
execute HTTP request: Received fatal alert: handshake_failure: Unable to
execute HTTP request: Received fatal alert: handshake_failure
ls: doesBucketExistV2 on hdfs: com.amazonaws.SdkClientException: Unable to
execute HTTP request: Received fatal alert: handshake_failure: Unable to
execute HTTP request: Received fatal alert: handshake_failure
```

Resolution: Make sure that JDK 11.x or later is installed and set to default the Java library. Refer to the Install Java package section for more information.

Scenario 2:

Failed to connect to StorageGRID with error message `Unable to find valid certification path to requested target`.

Reason: StorageGRID S3 endpoint server certificate is not trusted by Java program.

Sample error message:

```
[root@hdp6 ~]# hadoop fs -ls s3a://hdfs-test/
22/03/11 20:58:12 WARN impl.MetricsConfig: Cannot locate configuration:
tried hadoop-metrics2-s3a-file-system.properties,hadoop-
metrics2.properties
22/03/11 20:58:13 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot
period at 10 second(s).
22/03/11 20:58:13 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system started
22/03/11 20:58:13 INFO Configuration.deprecation: No unit for
fs.s3a.connection.request.timeout(0) assuming SECONDS
22/03/11 21:12:25 INFO impl.MetricsSystemImpl: Stopping s3a-file-system
metrics system...
22/03/11 21:12:25 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system stopped.
22/03/11 21:12:25 INFO impl.MetricsSystemImpl: s3a-file-system metrics
system shutdown complete.
22/03/11 21:12:25 WARN fs.FileSystem: Failed to initialize fileystem
s3a://hdfs-test/: org.apache.hadoop.fs.s3a.AWSClientIOException:
doesBucketExistV2 on hdfs: com.amazonaws.SdkClientException: Unable to
execute HTTP request: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find
valid certification path to requested target: Unable to execute HTTP
request: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find
valid certification path to requested target
```

Resolution: NetApp recommends using a server certificate issued by a known public certificate signing authority to make sure that the authentication is secure. Alternatively, add a custom CA or server certificate to the Java trust store.

Complete the following steps to add a StorageGRID custom CA or server certificate to the Java trust store.

1. Backup the existing default Java cacerts file.

```
cp -ap $JAVA_HOME/lib/security/cacerts
$JAVA_HOME/lib/security/cacerts.orig
```

2. Import the StorageGRID S3 endpoint cert to the Java trust store.

```
keytool -import -trustcacerts -keystore $JAVA_HOME/lib/security/cacerts
-storepass changeit -noprompt -alias sg-lb -file <StorageGRID CA or
server cert in pem format>
```

Troubleshooting tips

1. Increase the hadoop log level to DEBUG.

   ```
   export HADOOP_ROOT_LOGGER=hadoop.root.logger=DEBUG,console
   ```

2. Execute the command, and direct the log messages to error.log.

   ```
   hadoop fs -ls s3a://<bucket-name>/ &>error.log
   ```

By Angela Cheng

Use S3cmd to test and demonstrate S3 access on StorageGRID

By Aron Klein

S3cmd is a free command line tool and client for S3 operations. You can use s3cmd to to test and demonstrate s3 access on StorageGRID.

Install and configure S3cmd

To install S3cmd on a workstation or server, download it from command line S3 client. s3cmd is pre-installed on each StorageGRID node as a tool to aid in troubleshooting.

Initial configuration steps

1. s3cmd --configure
2. Provide only access_key and secret_key, for the the rest keep the defaults.
3. Test access with supplied credentials? [Y/n]: n (bypass the test as it will fail)
4. Save settings? [y/N] y
   a. Configuration saved to '/root/.s3cfg'
5. In .s3cfg make fields host_base and host_bucket empty after the "=" sign :
   a. host_base =
   b. host_bucket =

If you specify host_base and host_bucket in step 4, you don’t need to specify an endpoint with --host in the CLI. Example:

```
host_base = 192.168.1.91:8082
host_bucket = bucketX.192.168.1.91:8082
s3cmd ls s3://bucketX --no-check-certificate
```

Basic command examples

- **Create a bucket:**

```
s3cmd mb s3://s3cmdbucket --host=<endpoint>:<port> --no-check-certificate
```

- **List all buckets:**

```
s3cmd ls --host=<endpoint>:<port> --no-check-certificate
```

- **List all buckets and their contents:**

```
s3cmd la --host=<endpoint>:<port> --no-check-certificate
```

- **List objects in a specific bucket:**

```
s3cmd ls s3://<bucket> --host=<endpoint>:<port> --no-check-certificate
```

- **Delete a bucket:**

```
s3cmd rb s3://s3cmdbucket --host=<endpoint>:<port> --no-check-certificate
```

- **Put an object:**

```
s3cmd put <file> s3://<bucket> --host=<endpoint>:<port> --no-check-certificate
```

- **Get an object:**

```
s3cmd get s3://<bucket>/<object> <file> --host=<endpoint>:<port> --no-check
-certificate
```

- **Delete an object:**

```
s3cmd del s3://<bucket>/<object> --host=<endpoint>:<port> --no-check
-certificate
```

Vertica Eon mode database using NetApp StorageGRID as communal storage

By Angela Cheng

This guide describes the procedure to create a Vertica Eon Mode database with communal storage on NetApp StorageGRID.

Introduction

Vertica is an analytic database management software. It is a columnar storage platform designed to handle large volumes of data, which enables very fast query performance in a traditionally intensive scenario.
A Vertica database runs in one of the two modes: Eon or Enterprise. You can deploy both modes on-premises or in the cloud.

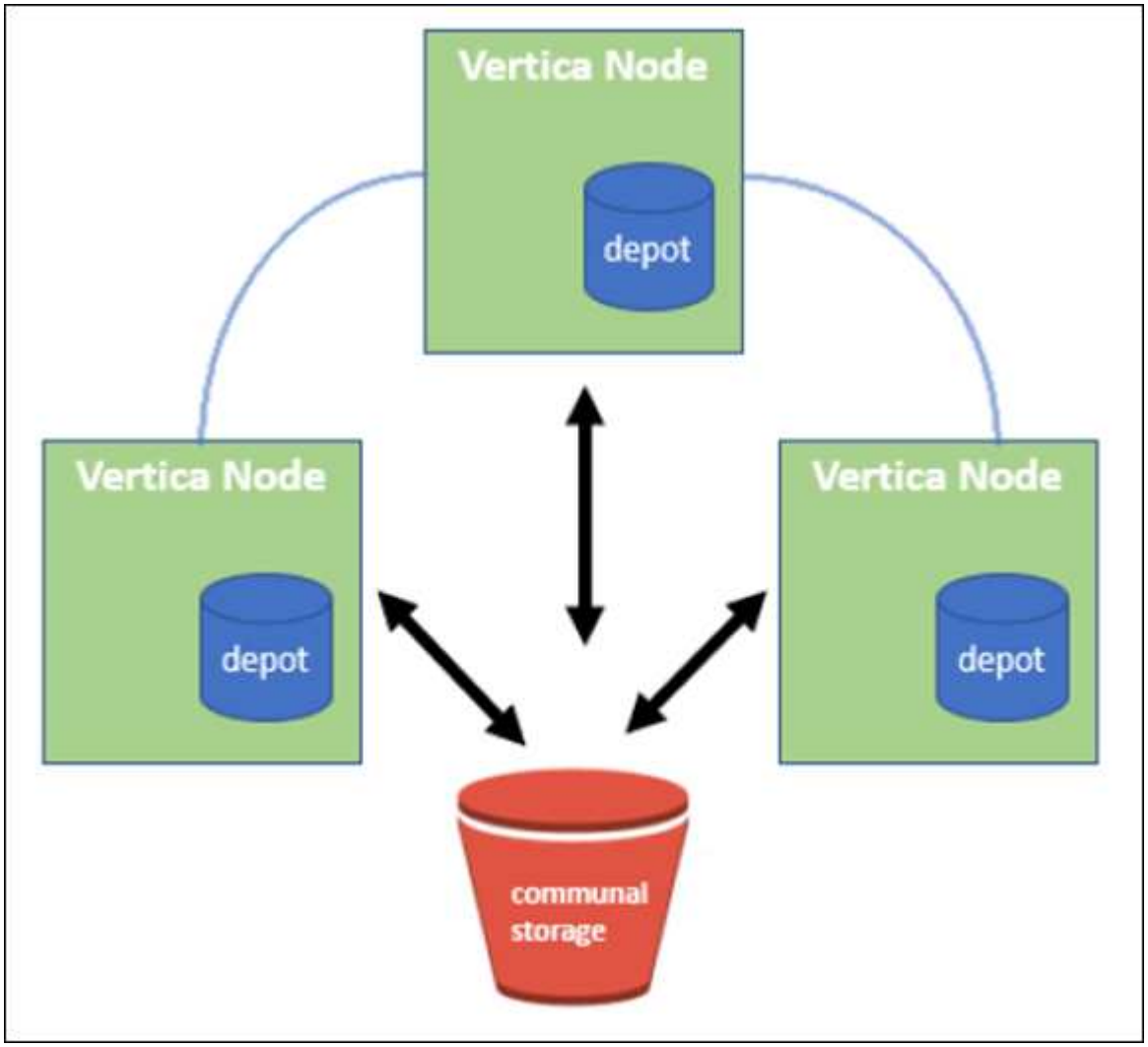
Eon and Enterprise modes primarily differ in where they store data:

- Eon Mode databases use communal storage for their data. This is recommended by Vertica.
- Enterprise Mode databases store data locally in the file system of nodes that make up the database.

Eon Mode architecture

Eon Mode separates the computational resources from the communal storage layer of the database, which allows the compute and storage to scale separately. Vertica in Eon Mode is optimized to address variable workloads and isolate them from one another by using separate compute and storage resources.

Eon Mode stores data in a shared object store called communal storage—an S3 bucket, either hosted on premises or on Amazon S3.



Communal storage

Instead of storing data locally, Eon Mode uses a single communal storage location for all data and the catalog (metadata). Communal storage is the database's centralized storage location, shared among the database nodes.

Communal storage has the following properties:

- Communal storage in the cloud or on-premises object storage is more resilient and less susceptible to data loss due to storage failures than storage on disk on individual machines.

- Any data can be read by any node using the same path.
- Capacity is not limited by disk space on nodes.
- Because data is stored communally, you can elastically scale your cluster to meet changing demands. If the data were stored locally on the nodes, adding or removing nodes would require moving significant amounts of data between nodes to either move it off nodes that are being removed, or onto newly created nodes.

The depot

One drawback of communal storage is its speed. Accessing data from a shared cloud location is slower than reading it from local disk. Also, the connection to communal storage can become a bottleneck if many nodes are reading data from it at once. To improve data access speed, the nodes in an Eon Mode database maintain a local disk cache of data called the depot. When executing a query, the nodes first check whether the data it needs is in the depot. If it is, then it finishes the query by using the local copy of the data. If the data is not in the depot, the node fetches the data from communal storage, and saves a copy in the depot.

NetApp StorageGRID recommendations

Vertica stores database data to object storage as thousands (or millions) of compressed objects (observed size is 200 to 500MB per object. When a user runs database queries, Vertica retrieves the selected range of data from these compressed objects in parallel using the byte-range GET call. Each byte-range GET is approximately 8KB.

During the 10TB database depot off user queries test, 4,000 to 10,000 GET (byte-range GET) requests per second were sent to the grid. When running this test using SG6060 appliances, though the CPU% utilization % per appliance node is low (around 20% to 30%), 2/3 of CPU time is waiting for I/O. A very small percentage (0% to 0.5%) of I/O wait is observed on the SGF6024.

Due to the high demand of small IOPS with very low latency requirements (the average should be less than 0.01 seconds), NetApp recommends using the SFG6024 for object storage services. If the SG6060 is needed for very large database sizes, the customer should work with the Vertica account team on depot sizing to support the actively queried dataset.

For the Admin Node and API Gateway Node, the customer can use the SG100 or SG1000. The choice depends on the number of users' query requests in parallel and database size. If the customer prefers to use a third-party load balancer, NetApp recommends a dedicated load balancer for high performance demand workload. For StorageGRID sizing, consult the NetApp account team.

Other StorageGRID configuration recommendations include:

- **Grid topology**. Do not mix the SGF6024 with other storage appliance models on the same grid site. If you prefer to use the SG6060 for long term archive protection, keep the SGF6024 with a dedicated grid load balancer in its own grid site (either physical or logical site) for an active database to enhance performance. Mixing different models of appliance on same site reduces the overall performance at the site.
- **Data protection**. Use replicate copies for protection. Do not use erasure coding for an active database. The customer can use erasure coding for long term protection of inactive databases.
- **Do not enable grid compression**. Vertica compresses objects before storing to object storage. Enabling grid compression does not further save storage usage and significantly reduces byte-range GET performance.
- **HTTP versus HTTPs S3 endpoint connection**. During the benchmark test, we observed about 5% performance improvement when using an HTTP S3 connection from the Vertica cluster to the StorageGRID load balancer endpoint. This choice should be based on customer security requirements.

Recommendations for a Vertica configuration include:

- **Vertica database default depot settings are enabled (value = 1) for read and write operations**. NetApp strongly recommends keeping these depot settings enabled to enhance performance.
- **Disable streaming limitations**. For configuration details, see the section Disabling streaming limitations.

Installing Eon Mode on-premises with communal storage on StorageGRID

The following sections describe the procedure, in order, to install Eon Mode on-premises with communal storage on StorageGRID. The procedure to configure on-premises Simple Storage Service (S3) compatible object storage is similar to the procedure in the Vertica guide, Install an Eon Mode Database on-premises.

The following setup was used for the functional test:

- StorageGRID 11.4.0.4
- Vertica 10.1.0
- Three virtual machines (VMs) with Centos 7.x OS for Vertica nodes to form a cluster. This setup is for the functional test only, not for the Vertica production database cluster.

These three nodes are set up with a Secure Shell (SSH) key to allow SSH without a password between the nodes within the cluster.

Information required from NetApp StorageGRID

To install Eon Mode on-premises with communal storage on StorageGRID, you must have the following prerequisite information.

- IP address or fully qualified domain name (FQDN) and port number of the StorageGRID S3 endpoint. If you are using HTTPS, use a custom certificate authority (CA) or self-signed SSL certificate implemented on the StorageGRID S3 endpoint.
- Bucket name. It must pre-exist and be empty.
- Access key ID and secret access key with read and write access to the bucket.

Creating an authorization file to access the S3 endpoint

The following prerequisites apply when creating an authorization file to access the S3 endpoint:

- Vertica is installed.
- A cluster is set up, configured, and ready for database creation.

To create an authorization file to access the S3 endpoint, follow these steps:

1. Log in to the Vertica node where you will run `admintools` to create the Eon Mode database.

   The default user is `dbadmin`, created during the Vertica cluster installation.

2. Use a text editor to create a file under the `/home/dbadmin` directory.
   The file name can be anything you want, for example, `sg_auth.conf`.
3. If the S3 endpoint is using a standard HTTP port 80 or HTTPS port 443, skip the port number. To use HTTPS, set the following values:

- awsenablehttps = 1, otherwise set the value to 0.
- awsauth = <s3 access key ID>:<secret access key>
- awsendpoint = <StorageGRID s3 endpoint>:<port>

  To use a custom CA or self-signed SSL certificate for the StorageGRID S3 endpoint HTTPS connection, specify the full file path and filename of the certificate. This file must be at the same location on each Vertica node and have read permission for all users. Skip this step if StorageGRID S3 Endpoint SSL certificate is signed by publicly known CA.

  - awscafile = <filepath/filename>

  For example, see the following sample file:

```
awsauth = MNVU4OYFAY2xyz123:03vuO4M4KmdfwffT8nqnBmnMVTr78Gu9wANabcxyz
awsendpoint = s3.england.connectlab.io:10443
awsenablehttps = 1
awscafile = /etc/custom-cert/grid.pem
```

  In a production environment, the customer should implement a server certificate signed by a publicly known CA on a StorageGRID S3 load balancer endpoint.

Choosing a depot path on all Vertica nodes

Choose or create a directory on each node for the depot storage path.
The directory you supply for the depot storage path parameter must have the following:

- The same path on all nodes in the cluster (for example, /home/dbadmin/depot)
- Be readable and writable by the dbadmin user
- Sufficient storage

  By default, Vertica uses 60% of the file system space containing the directory for depot storage. You can limit the size of the depot by using the --depot-size argument in the create_db command. See Sizing Your Vertica Cluster for an Eon Mode Database article for general Vertica sizing guidelines or consult with your Vertica account manager.

  The admintools create_db tool attempts to create the depot path for you if one does not exist.

Creating the Eon on-premises database

To create the Eon on-premises database, follow these steps:

1. To create the database, use the admintools create_db tool.

   The following list provides a brief explanation of arguments used in this example. See the Vertica document for a detailed explanation of all required and optional arguments.

   - -x <path/filename of authorization file created in “Creating an authorization file to access the S3 endpoint” >.

The authorization details are stored inside database after successful creation. You can remove this file to avoid exposing the S3 secret key.

- --communal-storage-location <s3://storagegrid bucketname>
- -s <comma-separated list of Vertica nodes to be used for this database>
- -d <name of database to be created>
- -p <password to be set for this new database>.
  For example, see the following sample command:

```
admintools -t create_db -x sg_auth.conf --communal-storage
-location=s3://vertica --depot-path=/home/dbadmin/depot --shard
-count=6 -s vertica-vm1,vertica-vm2,vertica-vm3 -d vmart -p
'<password>'
```

Creating a new database takes several minutes duration depending on number of nodes for the database. When creating database for the first time, you will be prompted to accept the License Agreement.

For example, see the following sample authorization file and `create db` command:

```
[dbadmin@vertica-vm1 ~]$ cat sg_auth.conf
awsauth = MNVU4OYFAY2CPKVXVxxxx:03vuO4M4KmdfwffT8nqnBmnMVTr78Gu9wAN+xxxx
awsendpoint = s3.england.connectlab.io:10445
awsenablehttps = 1

[dbadmin@vertica-vm1 ~]$ admintools -t create_db -x sg_auth.conf
--communal-storage-location=s3://vertica --depot-path=/home/dbadmin/depot
--shard-count=6 -s vertica-vm1,vertica-vm2,vertica-vm3 -d vmart -p
'xxxxxxxx'
Default depot size in use
Distributing changes to cluster.
    Creating database vmart
    Starting bootstrap node v_vmart_node0007 (10.45.74.19)
    Starting nodes:
        v_vmart_node0007 (10.45.74.19)
    Starting Vertica on all nodes. Please wait, databases with a large
catalog may take a while to initialize.
    Node Status: v_vmart_node0007: (DOWN)
    Node Status: v_vmart_node0007: (DOWN)
    Node Status: v_vmart_node0007: (DOWN)
    Node Status: v_vmart_node0007: (UP)
    Creating database nodes
    Creating node v_vmart_node0008 (host 10.45.74.29)
    Creating node v_vmart_node0009 (host 10.45.74.39)
    Generating new configuration information
```

```
    Stopping single node db before adding additional nodes.
    Database shutdown complete
    Starting all nodes
Start hosts = ['10.45.74.19', '10.45.74.29', '10.45.74.39']
    Starting nodes:
        v_vmart_node0007 (10.45.74.19)
        v_vmart_node0008 (10.45.74.29)
        v_vmart_node0009 (10.45.74.39)
    Starting Vertica on all nodes. Please wait, databases with a large
catalog may take a while to initialize.
    Node Status: v_vmart_node0007: (DOWN) v_vmart_node0008: (DOWN)
v_vmart_node0009: (DOWN)
    Node Status: v_vmart_node0007: (DOWN) v_vmart_node0008: (DOWN)
v_vmart_node0009: (DOWN)
    Node Status: v_vmart_node0007: (DOWN) v_vmart_node0008: (DOWN)
v_vmart_node0009: (DOWN)
    Node Status: v_vmart_node0007: (DOWN) v_vmart_node0008: (DOWN)
v_vmart_node0009: (DOWN)
    Node Status: v_vmart_node0007: (UP) v_vmart_node0008: (UP)
v_vmart_node0009: (UP)
Creating depot locations for 3 nodes
Communal storage detected: rebalancing shards

Waiting for rebalance shards. We will wait for at most 36000 seconds.
Installing AWS package
    Success: package AWS installed
Installing ComplexTypes package
    Success: package ComplexTypes installed
Installing MachineLearning package
    Success: package MachineLearning installed
Installing ParquetExport package
    Success: package ParquetExport installed
Installing VFunctions package
    Success: package VFunctions installed
Installing approximate package
    Success: package approximate installed
Installing flextable package
    Success: package flextable installed
Installing kafka package
    Success: package kafka installed
Installing logsearch package
    Success: package logsearch installed
Installing place package
    Success: package place installed
Installing txtindex package
    Success: package txtindex installed
```

```
Installing voltagesecure package
    Success: package voltagesecure installed
Syncing catalog on vmart with 2000 attempts.
Database creation SQL tasks completed successfully. Database vmart created
successfully.
```

Object size (byte)	Bucket/object key full path
61	s3://vertica/051/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a07/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a07_0_0.dfs
145	s3://vertica/2c4/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a3d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a3d_0_0.dfs
146	s3://vertica/33c/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a1d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a1d_0_0.dfs
40	s3://vertica/382/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a31/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a31_0_0.dfs
145	s3://vertica/42f/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a21/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a21_0_0.dfs
34	s3://vertica/472/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a25/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a25_0_0.dfs
41	s3://vertica/476/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a2d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a2d_0_0.dfs
61	s3://vertica/52a/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a5d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a5d_0_0.dfs

Object size (byte)	Bucket/object key full path
131	s3://vertica/5d2/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a19/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a19_0_0.dfs
91	s3://vertica/5f7/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a11/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a11_0_0.dfs
118	s3://vertica/82d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a15/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a15_0_0.dfs
115	s3://vertica/9a2/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a61/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a61_0_0.dfs
33	s3://vertica/acd/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a29/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a29_0_0.dfs
133	s3://vertica/b98/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a4d/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a4d_0_0.dfs
38	s3://vertica/db3/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a49/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a49_0_0.dfs
38	s3://vertica/eba/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a59/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a59_0_0.dfs
21521920	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000215e2/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000215e2.tar

Object size (byte)	Bucket/object key full path
6865408	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021602/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021602.tar
204217344	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021610/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021610.tar
16109056	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000217e0/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000217e0.tar
12853248	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021800/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021800.tar
8937984	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a000000002187a/026d63ae9d4a33237bf0e2c2cf2a794a00a000000002187a.tar
56260608	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000218b2/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000218b2.tar
53947904	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000219ba/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000219ba.tar
44932608	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000219de/026d63ae9d4a33237bf0e2c2cf2a794a00a00000000219de.tar
256306688	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a6e/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021a6e.tar

Object size (byte)	Bucket/object key full path
8062464	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021e34/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021e34.tar
20024832	s3://vertica/metadata/VMart/Libraries/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021e70/026d63ae9d4a33237bf0e2c2cf2a794a00a0000000021e70.tar
10444	s3://vertica/metadata/VMart/cluster_config.json
823266	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c13_13/chkpt_1.cat.gz
254	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c13_13/completed
2958	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c2_2/chkpt_1.cat.gz
231	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c2_2/completed
822521	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c4_4/chkpt_1.cat.gz
231	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c4_4/completed
746513	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_14_g14.cat

Object size (byte)	Bucket/object key full path
2596	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_3_g3.cat.gz
821065	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_4_g4.cat.gz
6440	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_5_g5.cat
8518	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_8_g8.cat
0	s3://vertica/metadata/VMart/nodes/v_vmart_node0016/Catalog/859703b06a3456d95d0be28575a673/tiered_catalog.cat
822922	s3://vertica/metadata/VMart/nodes/v_vmart_node0017/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c14_7/chkpt_1.cat.gz
232	s3://vertica/metadata/VMart/nodes/v_vmart_node0017/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c14_7/completed
822930	s3://vertica/metadata/VMart/nodes/v_vmart_node0017/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_14_g7.cat.gz
755033	s3://vertica/metadata/VMart/nodes/v_vmart_node0017/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_15_g8.cat
0	s3://vertica/metadata/VMart/nodes/v_vmart_node0017/Catalog/859703b06a3456d95d0be28575a673/tiered_catalog.cat
822922	s3://vertica/metadata/VMart/nodes/v_vmart_node0018/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c14_7/chkpt_1.cat.gz

Object size (byte)	Bucket/object key full path
232	s3://vertica/metadata/VMart/nodes/v_vmart_node0018/Catalog/859703b06a3456d95d0be28575a673/Checkpoints/c14_7/completed
822930	s3://vertica/metadata/VMart/nodes/v_vmart_node0018/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_14_g7.cat.gz
755033	s3://vertica/metadata/VMart/nodes/v_vmart_node0018/Catalog/859703b06a3456d95d0be28575a673/Txnlogs/txn_15_g8.cat
0	s3://vertica/metadata/VMart/nodes/v_vmart_node0018/Catalog/859703b06a3456d95d0be28575a673/tiered_catalog.cat

Disabling streaming limitations

This procedure is based on the Vertica guide for other on-premises object storage and should be applicable to StorageGRID.

1. After creating the database, disable the `AWSStreamingConnectionPercentage` configuration parameter by setting it to `0`.
   This setting is unnecessary for an Eon Mode on-premises installation with communal storage. This configuration parameter controls the number of connections to the object store that Vertica uses for streaming reads. In a cloud environment, this setting helps avoid having streaming data from the object store use up all the available file handles. It leaves some file handles available for other object store operations. Due to the low latency of on-premises object stores, this option is unnecessary.
2. Use a `vsql` statement to update the parameter value.
   The password is the database password that you set in “Creating the Eon on-premises database”.
   For example, see the following sample output:

```
[dbadmin@vertica-vm1 ~]$ vsql
Password:
Welcome to vsql, the Vertica Analytic Database interactive terminal.
Type:   \h or \? for help with vsql commands
        \g or terminate with semicolon to execute query
        \q to quit
dbadmin=> ALTER DATABASE DEFAULT SET PARAMETER
AWSStreamingConnectionPercentage = 0; ALTER DATABASE
dbadmin=> \q
```

Verifying depot settings

Vertica database default depot settings are enabled (value = 1) for read and write operations. NetApp strongly

recommends keeping these depot settings enabled to enhance performance.

```
vsql -c 'show current all;' | grep -i UseDepot
DATABASE | UseDepotForReads | 1
DATABASE | UseDepotForWrites | 1
```

Loading sample data (optional)

If this database is for testing and will be removed, you can load sample data to this database for testing. Vertica comes with sample dataset, VMart, found under `/opt/vertica/examples/VMart_Schema/` on each Vertica node.
You can find more information about this sample dataset here.

Follow these steps to load the sample data:

1. Log in as dbadmin to one of the Vertica nodes: cd /opt/vertica/examples/VMart_Schema/
2. Load sample data to the database and enter the database password when prompted in substeps c and d:
   a. `cd /opt/vertica/examples/VMart_Schema`
   b. `./vmart_gen`
   c. `vsql < vmart_define_schema.sql`
   d. `vsql < vmart_load_data.sql`
3. There are multiple predefined SQL queries, you can run some of them to confirm test data are loaded successfully into the database.
   For example: `vsql < vmart_queries1.sql`

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp StorageGRID 11.7 Product Documentation
- StorageGRID data sheet
- Vertica 10.1 Product Documentation

Version history

Version	Date	Document version history
Version 1.0	September 2021	Initial release.

By Angela Cheng

StorageGRID log analytics using ELK stack

By Angela Cheng

With the StorageGRID syslog forward feature, you can configure an external syslog server to collect and analyze StorageGRID log messages. ELK (Elasticsearch, Logstash, Kibana) has become one of the most popular log analytics solutions. Watch the StorageGRID log analysis using ELK video to view a sample ELK configuration and how it can be used to identify and troubleshoot failed S3 requests.
StorageGRID 11.9 supports exporting load balancer endpoint access log to external syslog server. Watch this Youtube Video to learn more about this new feature.
This article provides sample files of Logstash configuration, Kibana queries, charts and dashboard to give you a quick start for StorageGRID log management and analytics.

Requirements

- StorageGRID 11.6.0.2 or higher
- ELK (Elasticsearch, Logstash and Kibana) 7.1x or higher installed and in operation

Sample files

- Download the Logstash 7.x sample files package
  md5 checksum 148c23d0021d9a4bb4a6c0287464deab
  sha256 checksum f51ec9e2e3f842d5a7861566b167a561beb4373038b4e7bb3c8be3d522adf2d6
- Download the Logstash 8.x sample files package
  md5 checksum e11bae3a662f87c310ef363d0fe06835
  sha256 checksum 5c670755742cfdfd5aa723a596ba087e0153a65bcaef3934afdb682f61cd278d
- Download the Logstash 8.x sample files package for StorageGRID 11.9
  md5 checksum 41272857c4a54600f95995f6ed74800d
  sha256 checksum 67048ee8661052719990851e1ad960d4902fe537a6e135e8600177188da677c9

Assumption

Readers are familiar with StorageGRID and ELK terminology and operations.

Instruction

Two sample versions are provided due to differences in names defined by grok patterns.
For example, the SYSLOGBASE grok pattern in Logstash config file defines field names differently depending on the installed Logstash version.

```
match => {"message" => '<%{POSINT:syslog_pri}>%{SYSLOGBASE}
%{GREEDYDATA:msg-details}'}
```

Logstash 7.17 sample

Field	Value
t _id	7C1MaYEBRH8UbfKnIls8
t _index	sgrid2-2022.06.15
# _score	-
t _type	_doc
@timestamp	Jun 15, 2022 @ 17:36:46.038
t host	grid2-site2-s1
t logsource	SITE2-S1
t msg-details	Reloading syslog service
t pid	628
t program	update-sysl
t syslog_pri	37
t timestamp	Jun 15 21:36:46

Logstash 8.23 sample

Table JSON

Search field names

Actions	Field	Value
	_id	yuhOiIEBVP6KX4EwqcyU
	_index	sglog-2022.06.21
	# _score	-
	@timestamp	Jun 21, 2022 @ 18:07:45.444
	t event.original	<28>Jun 21 22:07:45 SITE2-S3 ADE: syslog messages being dropped
	t host.hostname	SITE2-S3
	t msg-details	syslog messages being dropped
	t process.name	ADE
	t syslog_pri	28
	t timestamp	Jun 21 22:07:45

Steps

1. Unzip the provided sample based on your installed ELK version.
   The sample folder includes two Logstash config samples:
   sglog-2-file.conf: this config file outputs StorageGRID log messages to a file on Logstash without data transformation. You can use this to confirm Logstash is receiveing StorageGRID messages or to help understand StorageGRID log patterns.
   sglog-2-es.conf: this config file transforms StorageGRID log messages using various pattern and filters. It includes example drop statements, which drop messages based on patterns or filter. The output is sent to Elasticsearch for indexing.
   Customize the selected config file according to the instruction inside the file.
2. Test the customized config file:

   ```
   /usr/share/logstash/bin/logstash --config.test_and_exit -f <config-file-path/file>
   ```

   If the last line returned is similar to the below line, the config file has no syntax errors:

   ```
   [LogStash::Runner] runner - Using config.test_and_exit mode. Config Validation Result: OK. Exiting Logstash
   ```

3. Copy the customized conf file to the Logstash server's config: /etc/logstash/conf.d
   If you have not enabled config.reload.automatic in /etc/logstash/logstash.yml, restart the Logstash service. Otherwise, wait for the config reload interval to elapse.

   ```
    grep reload /etc/logstash/logstash.yml
   # Periodically check if the configuration has changed and reload the pipeline
   config.reload.automatic: true
   config.reload.interval: 5s
   ```

4. Check /var/log/logstash/logstash-plain.log and confirm there are no errors starting Logstash with the new config file.
5. Confirm TCP port is started and listening.
   In this example, TCP port 5000 is used.

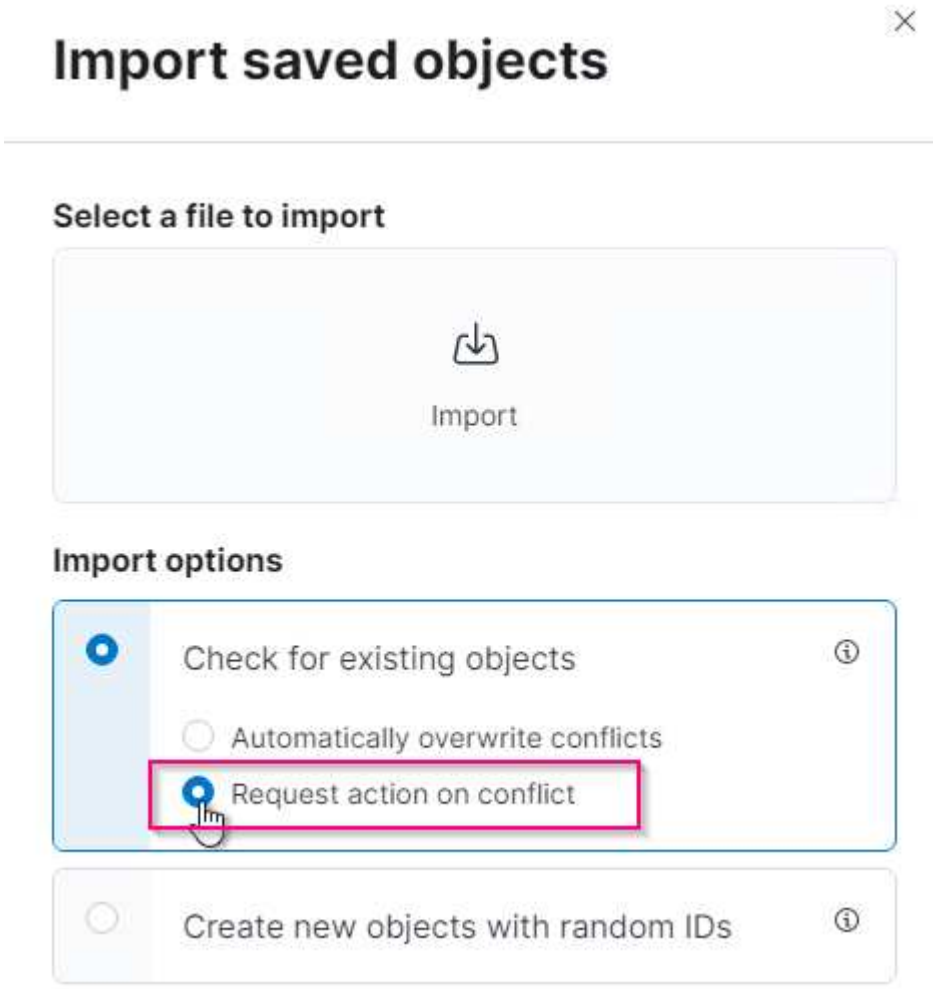
   ```
   netstat -ntpa | grep 5000
   tcp6       0      0 :::5000                 :::*
   LISTEN      25744/java
   ```

6. From the StorageGRID manager GUI, configure external syslog server to send log messages to Logstash. Refer to the demo video for details.
7. You need to configure or disable firewall on the Logstash server to allow StorageGRID nodes connection to the defined TCP port.

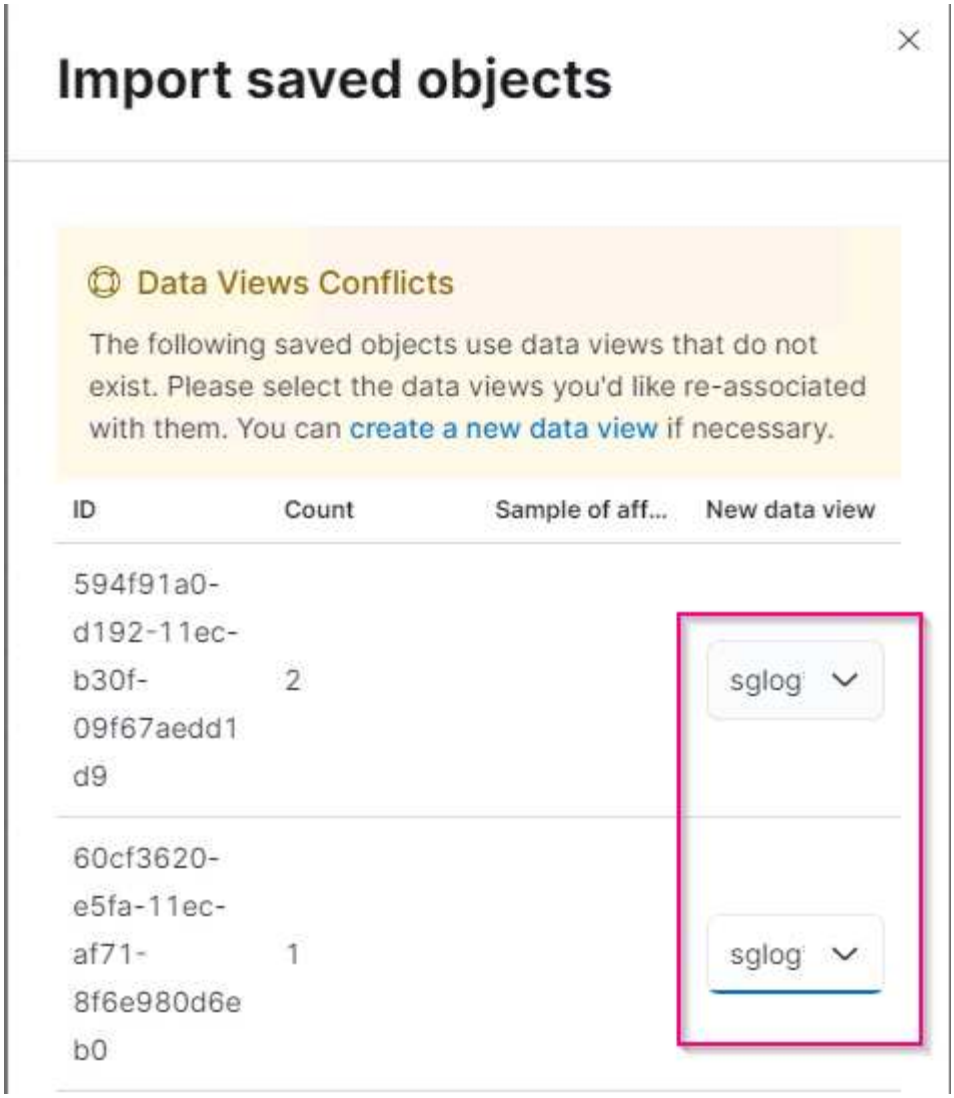
8. From Kibana GUI, select Management → Dev Tools. On the Console page, run this GET command to confirm new indices are created on Elasticsearch.

```
GET /_cat/indices/*?v=true&s=index
```

9. From Kibana GUI, create index pattern (ELK 7.x) or data view (ELK 8.x).
10. From Kibana GUI, enter 'saved objects' in the search box which is located in the top center.
On the Saved Objects page, select Import. Under Import options, select 'Request action on conflict'



Import elk<version>-query-chart-sample.ndjson.
When prompted to resolve the conflict, select the index pattern or data view you created in step 8.



The following Kibana objects are imported:
Query
* audit-msg-s3rq-orlm
* bycast log s3 related messages
* loglevel warning or above
* failed security event
* nginx-gw endpoint access log (available only in elk8-sample-for-sg119.zip)
Chart
* s3 requests count based on bycast.log
* HTTP status code
* audit msg breakdown by type
* average s3 response time
Dashboard
* S3 request dashboard using the above charts.

You are now ready to perform StorageGRID log analysis using Kibana.

Additional resources

- syslog101

- What is the ELK stack
- Grok patterns list
- A beginner's guide to Logstash: Grok
- A practical guide to Logstash: syslog deep dive
- Kibana guide – Explore the document
- StorageGRID audit log messages reference

Use Prometheus and Grafana to extend your metrics retention

By Aron Klein

This technical report provides detailed instructions for configuring NetApp StorageGRID with external Prometheus and Grafana services.

Introduction

StorageGRID stores metrics using Prometheus and provides visualizations of these metrics through built in Grafana dashboards. The Prometheus metrics can be accessed securely from StorageGRID by configuring client access certificates and enabling prometheus access for the specified client. Today, the retention of this metric data is limited by the storage capacity of the administration node. To gain a longer duration and an ability to create customized visualizations of these metrics we will deploy a new Prometheus and Grafana server, configure our new server to scrape the metrics from StorageGRIDs instance, and build a dashboard with the metrics that are important to us. You can get more information on the Prometheus metrics collected in the StorageGRID documentation.

Federate Prometheus

Lab details

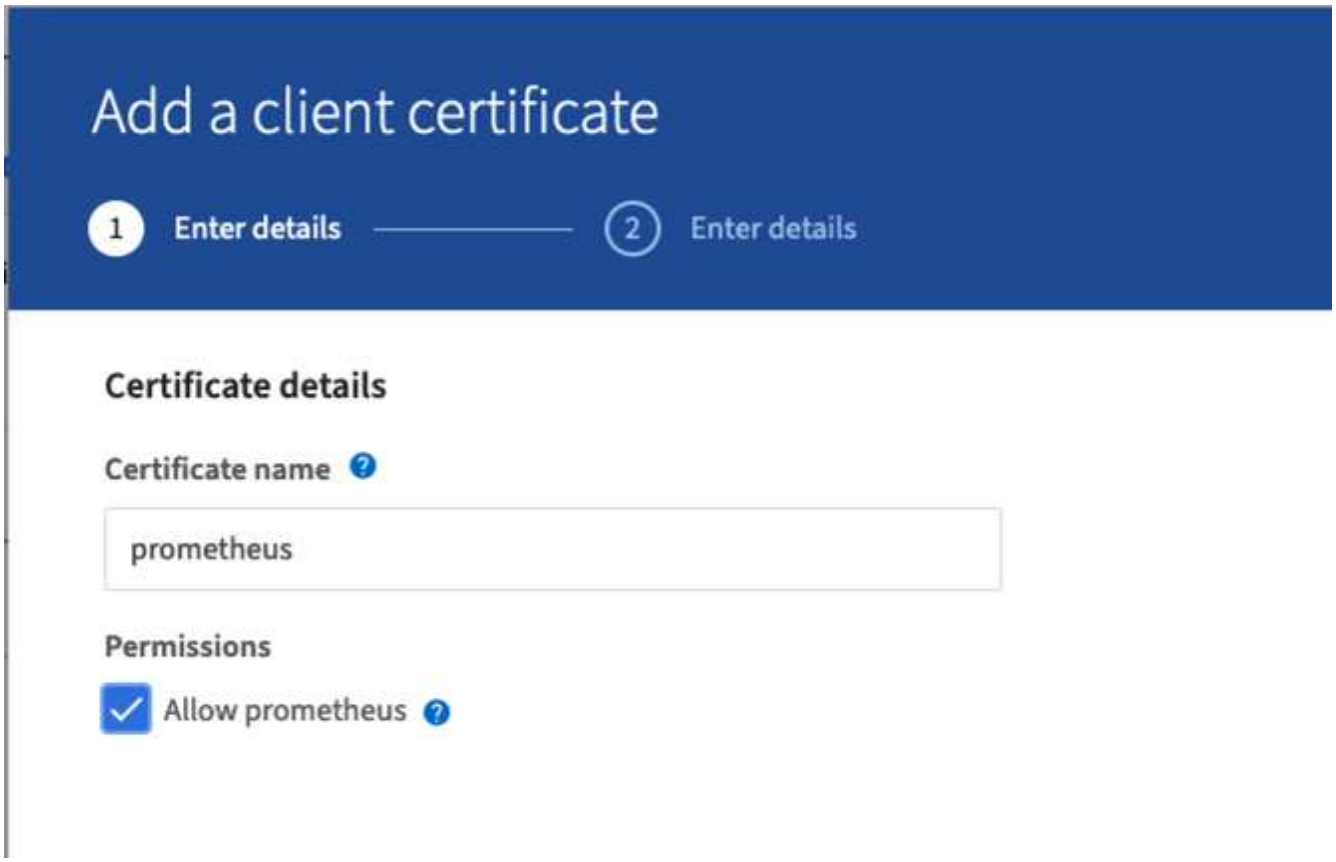
For the purposes of this example, I will be using all virtual machines for StorageGRID 11.6 nodes, and a Debian 11 server. The StorageGRID management interface is configured with a publicly trusted CA certificate. This example will not go through the installation and configuration of the StorageGRID system or Debian linux installation. You can use any Linux flavor you wish that is supported by Prometheus and Grafana. Both Prometheus and Grafana can install as docker containers, build from source, or pre-compiled binaries. In this example I will be installing both Prometheus and Grafana binaries directly on the same Debian server. Download and follow the basic installation instructions from https://prometheus.io and https://grafana.com/grafana/ respectively.

Configure StorageGRID for Prometheus Client access

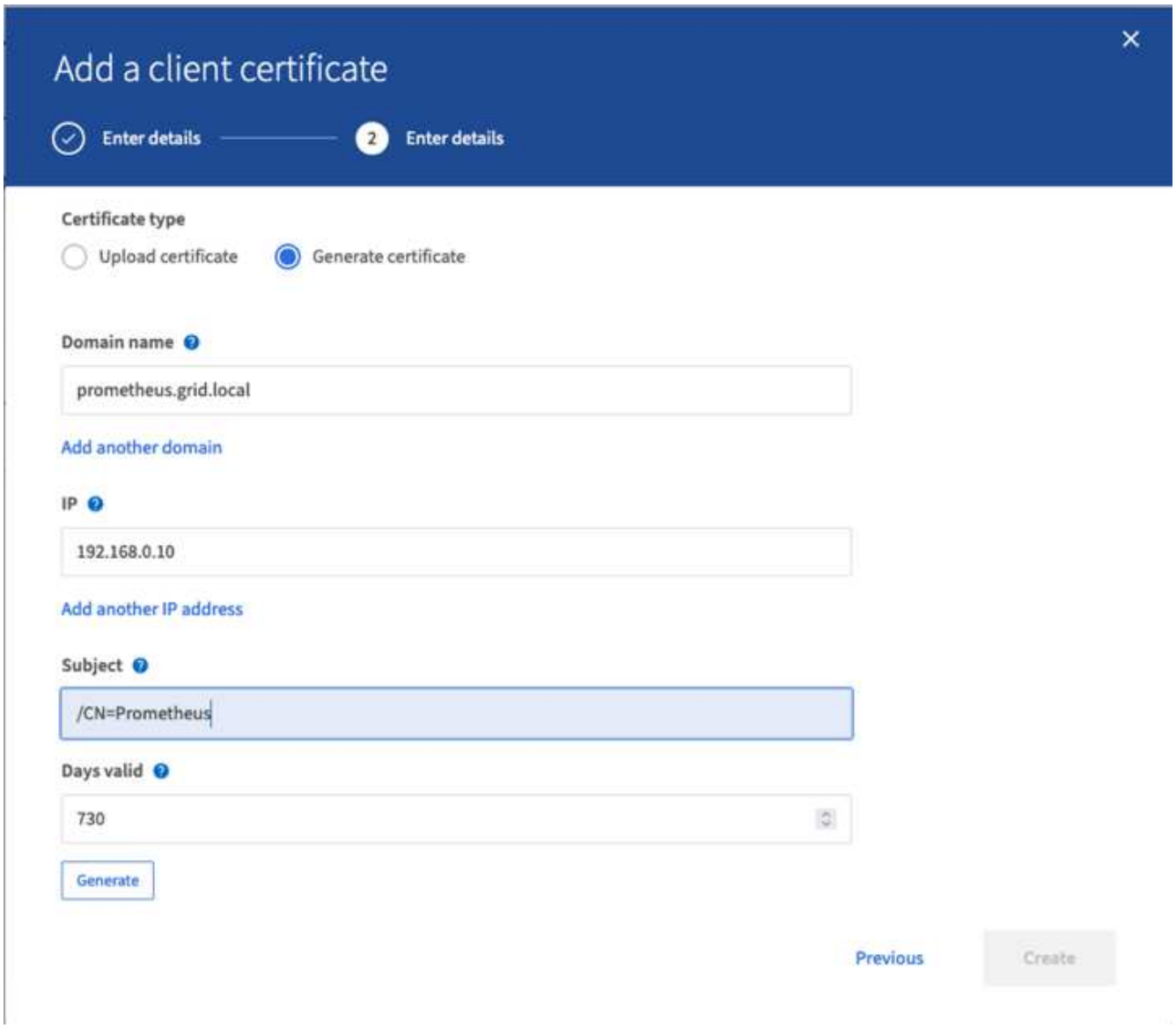
In order to gain access to StorageGRIDs stored prometheus metrics you must generate or upload a client certificate with private key, and enable permission for the client. The StorageGRID managament interface must have an SSL certificate. This certificate must be trusted by the prometheus server either by a trusted CA, or manually trusted if it is self-signed. To read more, please visit the StorageGRID documentation.

1. In the StorageGRID management interface, select "CONFIGURATION" on the bottom left hand side, and in the second column under "Security" click on Certificates.
2. On the Certificates page select the "Client" tab and click on the "Add" button.

3. Provide a name for the client that will be granted access and use this certificate. Click on the box under "Permissions", in front of "Allow Prometheus" and click the Continue button.

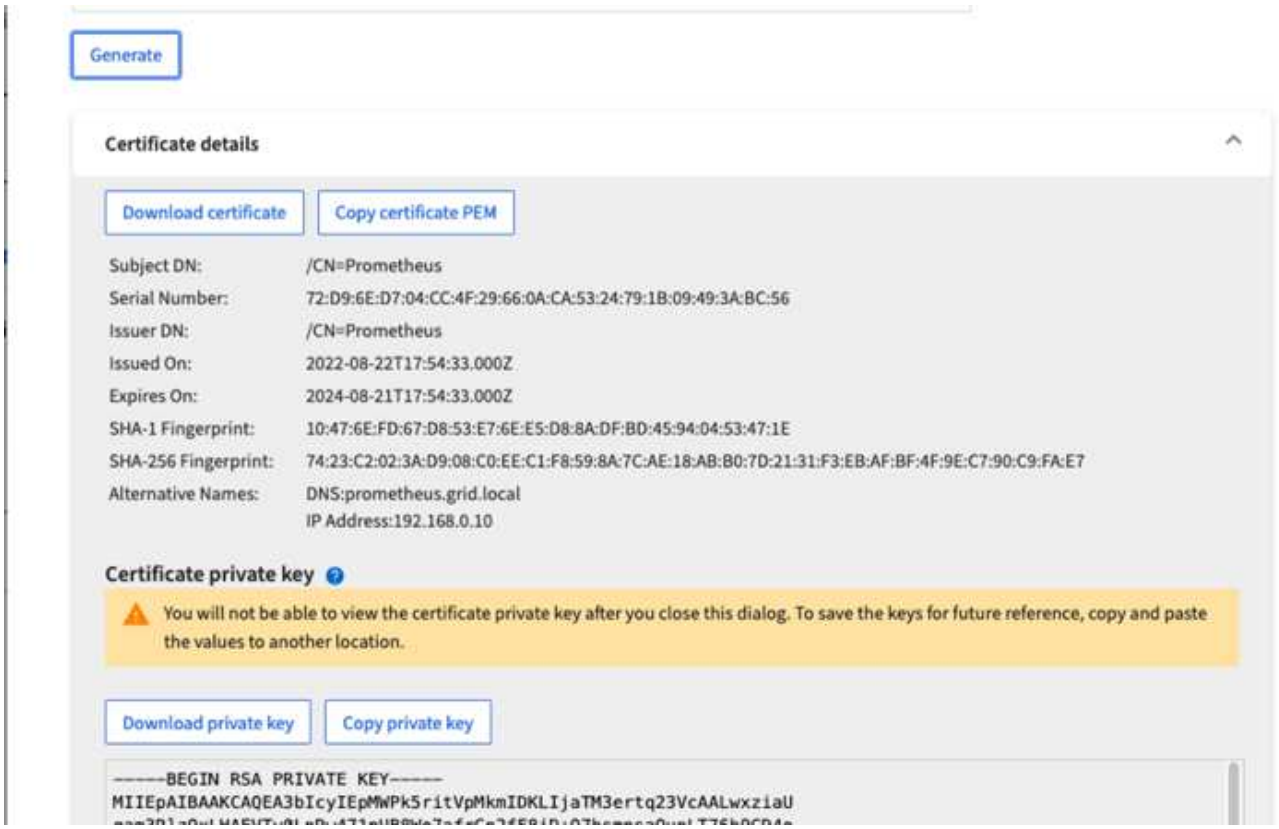


4. If you have a CA signed certificate you can select the radio button for "Upload certificate", but in our case we are going to let storageGRID generate the client certificate by selecting the radio button for "Generate Certificate". The required fields will be displayed to be filled in. Enter the FQDN for the client server, the IP of the server, the subject, and Days valid. Then click the "Generate" button.



> Be mindful of the certificate days valid entry as you will need to renew this certificate in both StorageGRID and the Prometheus server before it expires to maintain uninterrupted collection.

1. Download the certificate pem file, and the private key pem file.



This is the only time you can download the private key, so make sure you do not skip this step.

Prepare the Linux server for Prometheus installation

Before installing Prometheus, I want to get my environment prepared with a Prometheus user, the directory structure, and configure the capacity for the metrics storage location.

1. Create the Prometheus user.

```
sudo useradd -M -r -s /bin/false Prometheus
```

2. Create the directories for Prometheus, client certificate, and metrics data.

```
sudo mkdir /etc/Prometheus /etc/Prometheus/cert /var/lib/Prometheus
```

3. I formatted the disk I am using for metrics retention with an ext4 filesystem.

```
mkfs -t ext4 /dev/sdb
```

4. I then mounted the filesystem to the Prometheus metrics directory.

```
sudo mount -t auto /dev/sdb /var/lib/prometheus/
```

5. Obtain the uuid of the disk you are using for your metrics data.

```
sudo ls -al /dev/disk/by-uuid/
   lrwxrwxrwx 1 root root   9 Aug 18 17:02 9af2c5a3-bfc2-4ec1-85d9-
ebab850bb4a1 -> ../../sdb
```

6. Adding an entry in /etc/fstab/ making the mount persist across reboots using the uuid of /dev/sdb.

```
/etc/fstab
UUID=9af2c5a3-bfc2-4ec1-85d9-ebab850bb4a1 /var/lib/prometheus   ext4
defaults    0   0
```

Install and configure Prometheus

Now that the server is ready, I can begin the Prometheus installation and configure the service.

1. Extract the Prometheus installation package

```
tar xzf prometheus-2.38.0.linux-amd64.tar.gz
```

2. Copy the binaries to /usr/local/bin and change the ownership to the prometheus user created earlier

```
sudo cp prometheus-2.38.0.linux-amd64/{prometheus,promtool}
/usr/local/bin
sudo chown prometheus:prometheus /usr/local/bin/{prometheus,promtool}
```

3. Copy the consoles and libraries to /etc/prometheus

```
sudo cp -r prometheus-2.38.0.linux-amd64/{consoles,console_libraries}
/etc/prometheus/
```

4. Copy the client certificate and private key pem files downloaded earlier from StorageGRID to /etc/prometheus/certs
5. Create the prometheus configuration yaml file

```
sudo nano /etc/prometheus/prometheus.yml
```

6. Insert the following configuration. The job name can be anything you wish. Change the "-targets: ['']" to the

FQDN of the admin node, and if you altered the names of the certificate and private key file names, please update the tls_config section to match. then save the file. If your grid management interface, is using a self-signed certificate, download the certificate and place it with the client certificate with a unique name, and in the tls_config section add ca_file: /etc/prometheus/cert/UIcert.pem

a. In this example I am collecting all of the metrics that begin with alertmanager, cassandra, node, and storagegrid. You can see more information on the Prometheus metrics in the StorageGRID documentation.

```
# my global config
global:
  scrape_interval: 60s # Set the scrape interval to every 15 seconds.
Default is every 1 minute.

scrape_configs:
  - job_name: 'StorageGRID'
    honor_labels: true
    scheme: https
    metrics_path: /federate
    scrape_interval: 60s
    scrape_timeout: 30s
    tls_config:
      cert_file: /etc/prometheus/cert/certificate.pem
      key_file: /etc/prometheus/cert/private_key.pem
    params:
      match[]:
        -
'{__name__=~"alertmanager_.*|cassandra_.*|node_.*|storagegrid_.*"}'
    static_configs:
    - targets: ['sgdemo-rtp.netapp.com:9091']
```

If your grid management interface is using a self-signed certificate, download the certificate and place it with the client certificate with a unique name. In the tls_config section add the certificate above the client certificate and private key lines

```
        ca_file: /etc/prometheus/cert/UIcert.pem
```

1. Change the ownership of all files and directories in /etc/prometheus, and /var/lib/prometheus to the prometheus user

```
sudo chown -R prometheus:prometheus /etc/prometheus/
sudo chown -R prometheus:prometheus /var/lib/prometheus/
```

2. Create a prometheus service file in /etc/systemd/system

```
sudo nano /etc/systemd/system/prometheus.service
```

3. Insert the following lines, note the --storage.tsdb.retention.time=1y which sets the retention of the metric data to 1 year. Alternatively, you could use --storage.tsdb.retention.size=300GiB to base retention on storage limits. This is the only location to set the metrics retention.

```
[Unit]
Description=Prometheus Time Series Collection and Processing Server
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
        --config.file /etc/prometheus/prometheus.yml \
        --storage.tsdb.path /var/lib/prometheus/ \
        --storage.tsdb.retention.time=1y \
        --web.console.templates=/etc/prometheus/consoles \
        --web.console.libraries=/etc/prometheus/console_libraries

[Install]
WantedBy=multi-user.target
```

4. Reload the systemd service to register the new prometheus service. then start and enable the prometheus service.

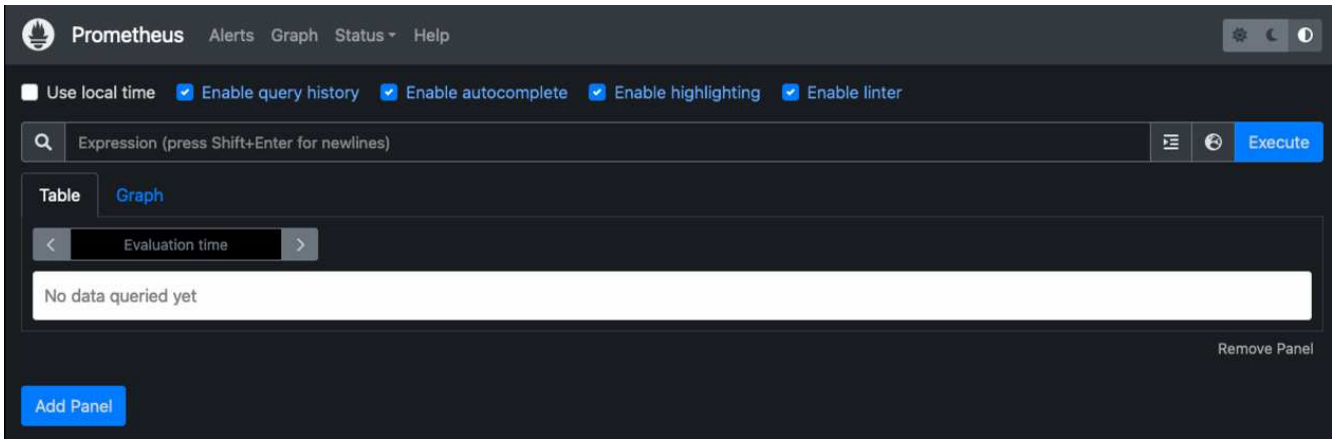
```
sudo systemctl daemon-reload
sudo systemctl start prometheus
sudo systemctl enable prometheus
```

5. Check the service is runing properly

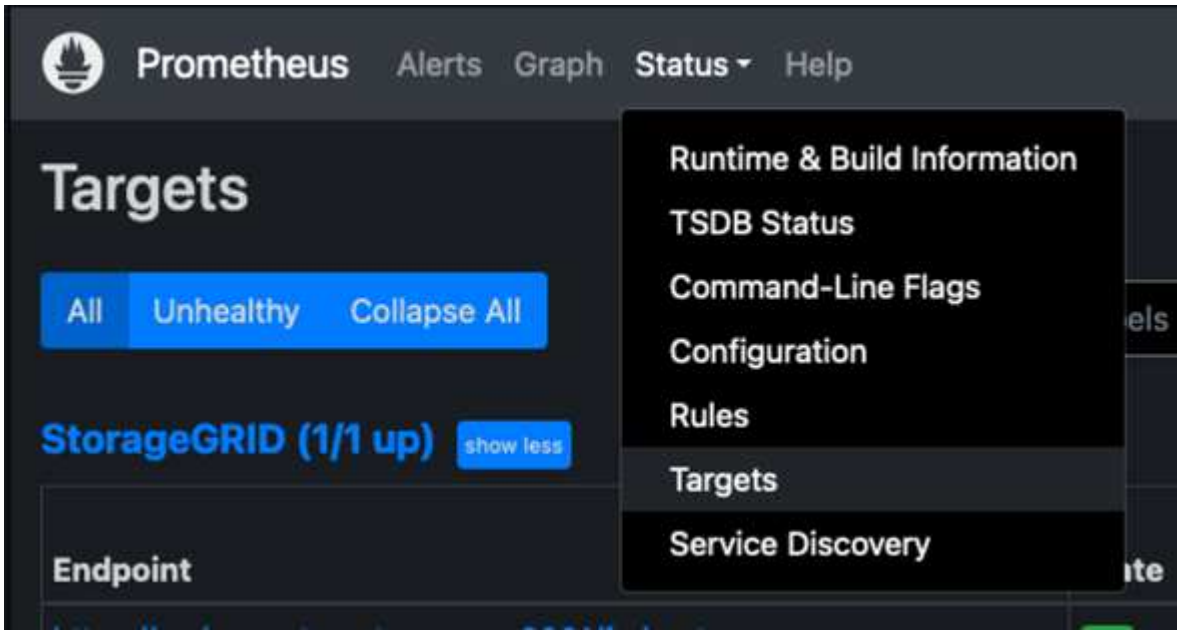
```
sudo systemctl status prometheus
```

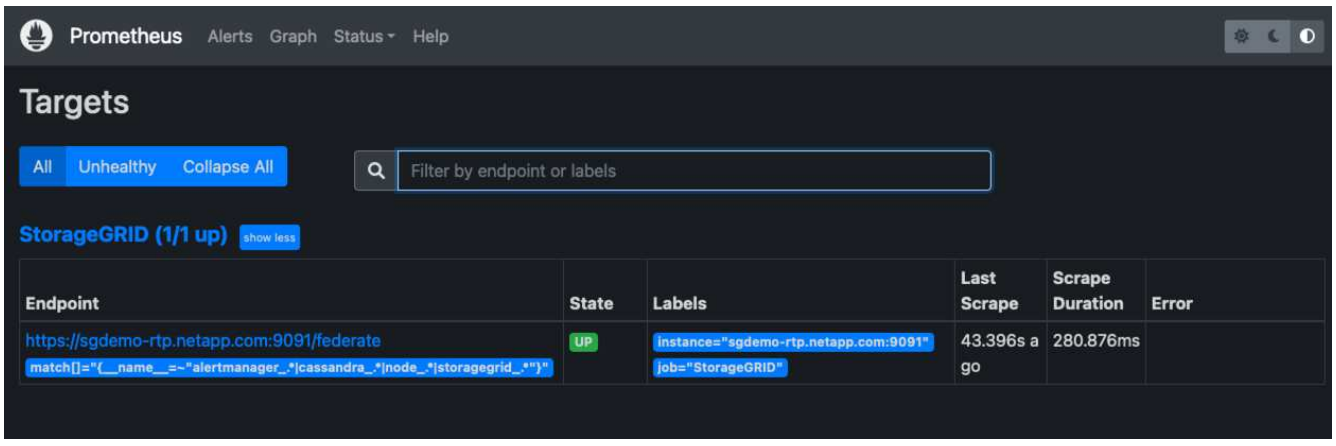
```
● prometheus.service - Prometheus Time Series Collection and Processing
Server
     Loaded: loaded (/etc/systemd/system/prometheus.service; enabled;
vendor preset: enabled)
     Active: active (running) since Mon 2022-08-22 15:14:24 EDT; 2s ago
   Main PID: 6498 (prometheus)
      Tasks: 13 (limit: 28818)
     Memory: 107.7M
        CPU: 1.143s
     CGroup: /system.slice/prometheus.service
             └─6498 /usr/local/bin/prometheus --config.file
/etc/prometheus/prometheus.yml --storage.tsdb.path /var/lib/prometheus/
--web.console.templates=/etc/prometheus/consoles --web.con>

Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.510Z caller=head.go:544 level=info component=tsdb
msg="Replaying WAL, this may take a while"
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.816Z caller=head.go:615 level=info component=tsdb msg="WAL
segment loaded" segment=0 maxSegment=1
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.816Z caller=head.go:615 level=info component=tsdb msg="WAL
segment loaded" segment=1 maxSegment=1
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.816Z caller=head.go:621 level=info component=tsdb msg="WAL
replay completed" checkpoint_replay_duration=55.57µs wal_rep>
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.831Z caller=main.go:997 level=info fs_type=EXT4_SUPER_MAGIC
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.831Z caller=main.go:1000 level=info msg="TSDB started"
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.831Z caller=main.go:1181 level=info msg="Loading
configuration file" filename=/etc/prometheus/prometheus.yml
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.832Z caller=main.go:1218 level=info msg="Completed loading
of configuration file" filename=/etc/prometheus/prometheus.y>
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.832Z caller=main.go:961 level=info msg="Server is ready to
receive web requests."
Aug 22 15:14:24 aj-deb-prom01 prometheus[6498]: ts=2022-08-
22T19:14:24.832Z caller=manager.go:941 level=info component="rule
manager" msg="Starting rule manager..."
```

6. You should now be able to browse to the UI of your prometheus server http://Prometheus-server:9090 and see the UI

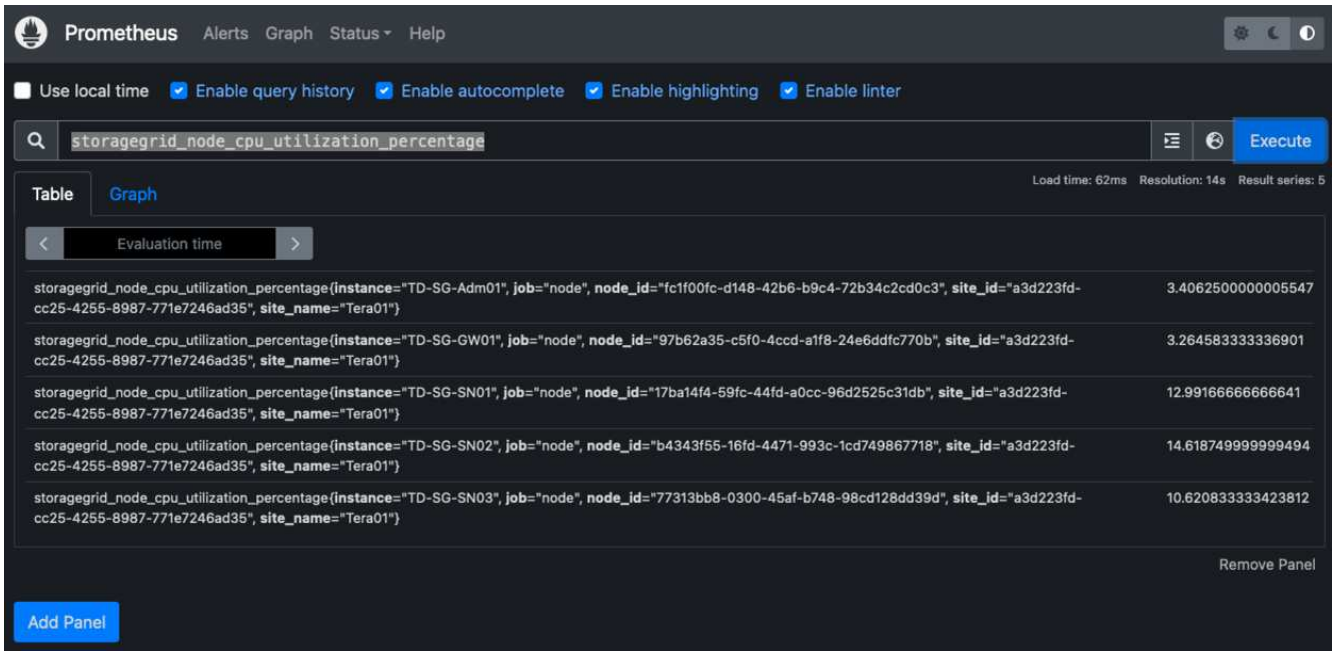


7. Under "Status" Targets you can see the status of the StorageGRID endpoint we configured in prometheus.yml





8. On the Graph page, you can execute a test query and verify the data is successfully being scraped. for example enter "storagegrid_node_cpu_utilization_percentage" into the query bar and click the Execute button.



Install and configure Grafana

Now that prometheus is installed and working, we can move on to installing Grafana and configuring a dashboard

Grafana Instalation

1. Install the latest enterprise edition of Grafana

```
sudo apt-get install -y apt-transport-https
sudo apt-get install -y software-properties-common wget
sudo wget -q -O /usr/share/keyrings/grafana.key
https://packages.grafana.com/gpg.key
```

2. Add this repository for stable releases:

```
echo "deb [signed-by=/usr/share/keyrings/grafana.key]
https://packages.grafana.com/enterprise/deb stable main" | sudo tee -a
/etc/apt/sources.list.d/grafana.list
```

3. After you add the repository.

```
sudo apt-get update
sudo apt-get install grafana-enterprise
```

4. Reload the systemd service to register the new grafana service. then start and enable the Grafana service.

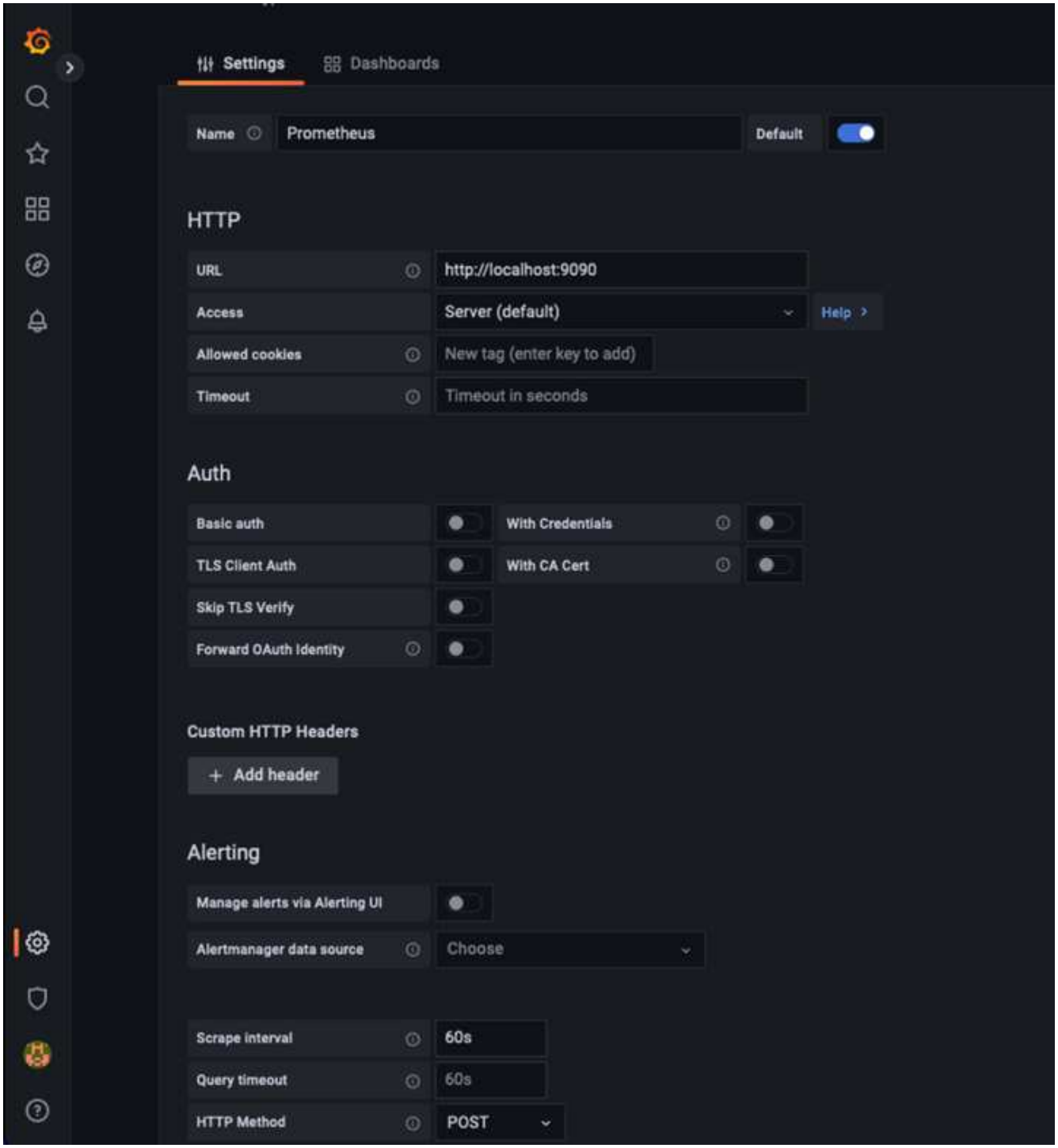
```
sudo systemctl daemon-reload
sudo systemctl start grafana-server
sudo systemctl enable grafana-server.service
```

5. Grafana is now installed and running. When you open a browser to HTTP://Prometheus-server:3000 you will be greeted with the Grafana login page.
6. The default login credentials are admin/admin, and you should set a new password as it prompts you to.

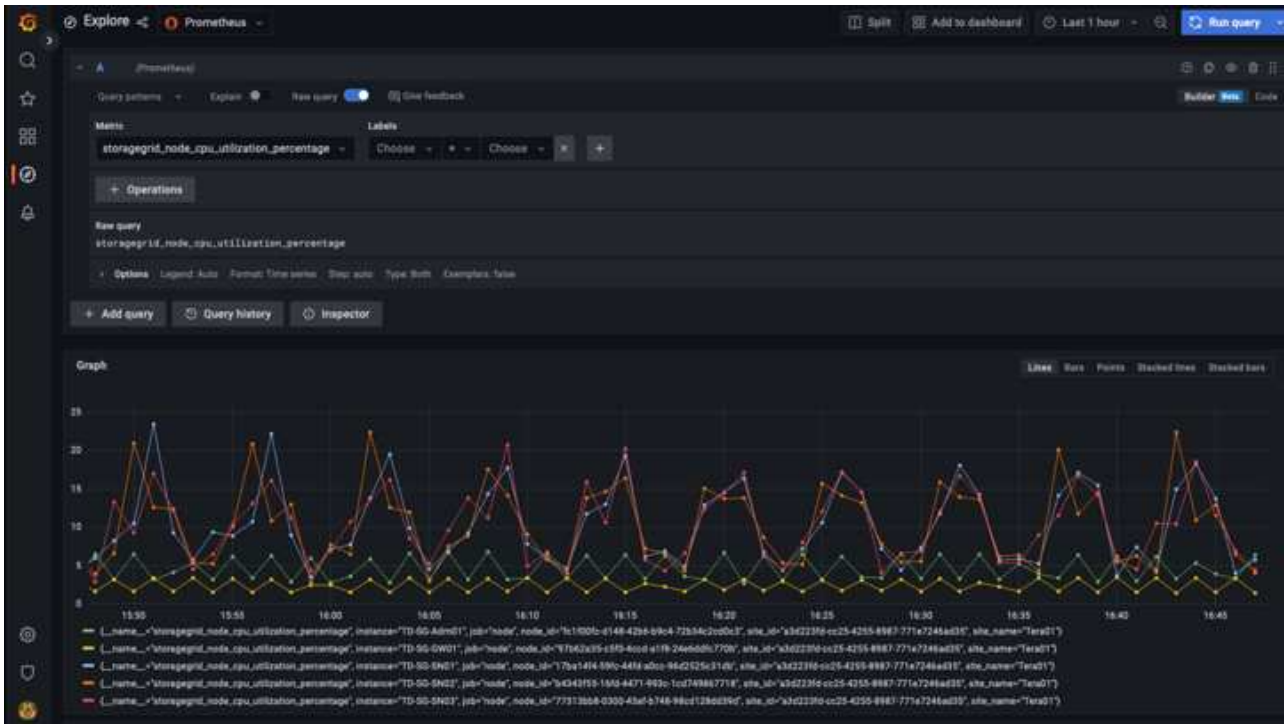
Create a Grafana dashboard for StorageGRID

With Grafana and Prometheus installed and running, now its time to connect the two by creating a data source and build a dashboard

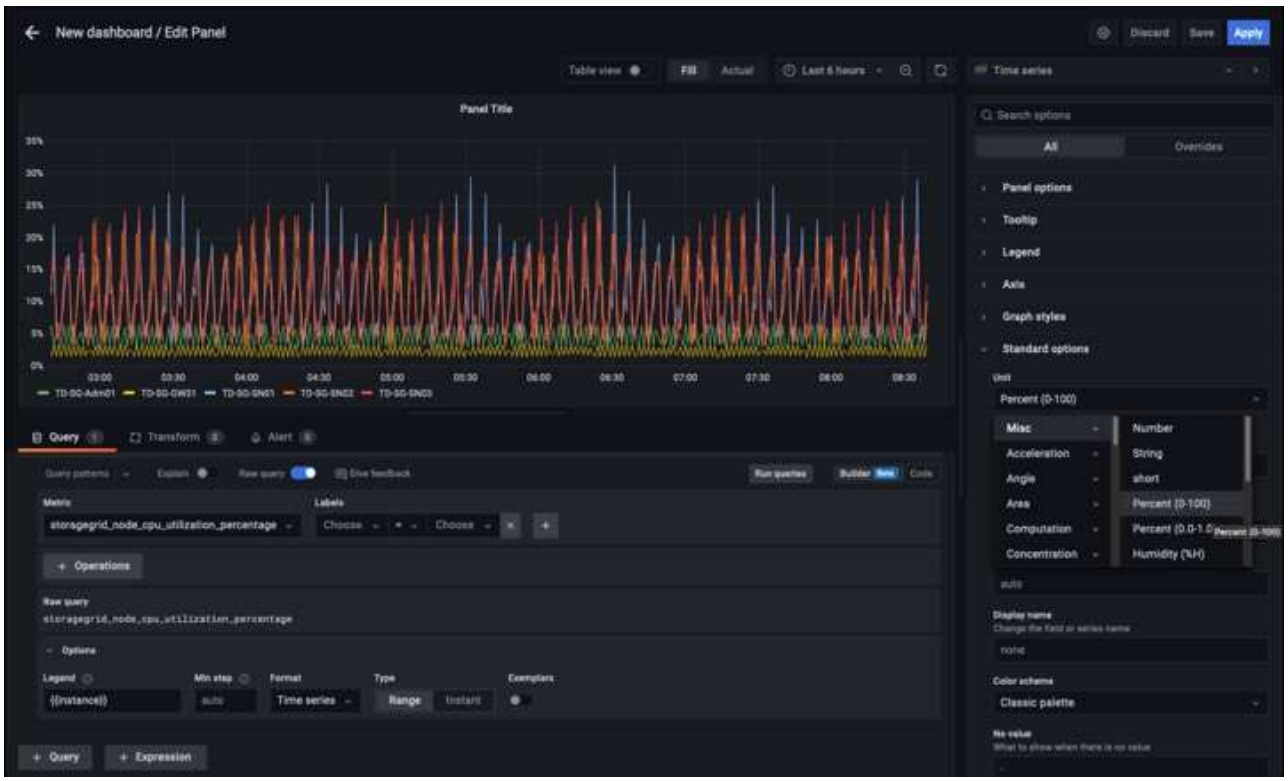
1. On the left hand pane expand "Configuration" and select "Data sources", then click on the "Add Data source" button
2. Prometheus will be one of the top data sources to choose from. If it is not, then use the search bar to locate "Prometheus"
3. Configure the Prometheus source by entering the URL of the prometheus instance, and the scrape interval to match the Prometheus interval. I also disabled the alerting section as I did not configure the alert manager on prometheus.



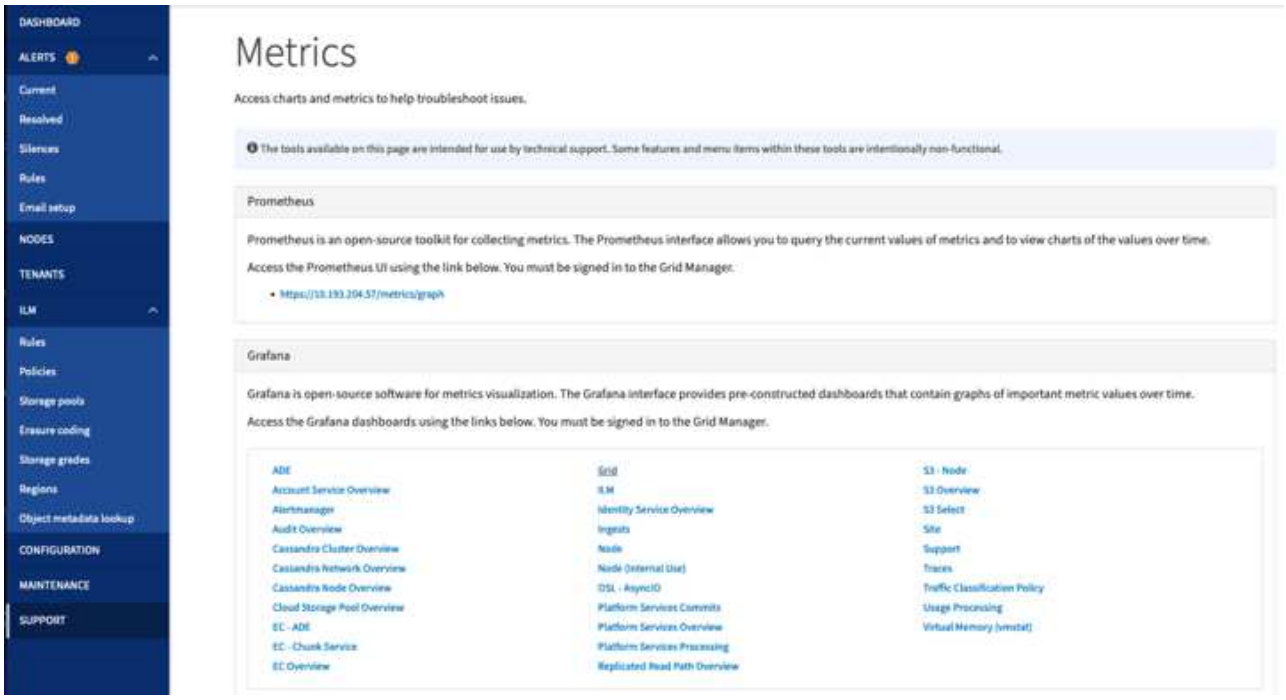
4. With the desired settings entered, scroll down to the bottom and click on "Save & test"
5. After the configuration test is successful, click on the explore button.
    a. In the explore window you can use the same metric we tested Prometheus with "storagegrid_node_cpu_utilization_percentage", and click the "Run query" button



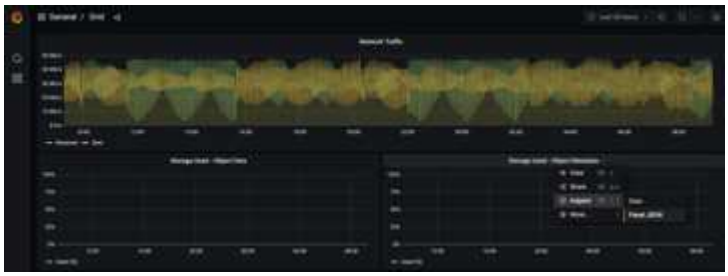
6. Now that we have the data source configured, we can create a dashboard.
   a. On the left hand pane expand "Dashboards", and select "+ new Dashboard"
   b. Select "Add a new panel"
   c. Configure the new panel by selecting a metric, again I will use "storagegrid_node_cpu_utilization_percentage", Enter a title for the panel, expand "Options" at the bottom and for legend change to custom and enter "{{instance}}" to define the node names", and on the right pane under "Standard options" set "Unit" to "Misc/Percent(0-100)". Then click "Apply" to save the panel to the dashboard.



7. We could continue to build out our dashboard like this for each metric we want, but luckily StorageGRID already has dashboards with panels we can copy into our custom dashboards.
   a. From the StorageGRID management interface left hand pane, select "Support", and at the bottom of the "Tools" column click on "Metrics".
   b. Within metrics, I am going to select the "Grid" link on the top of the middle column.

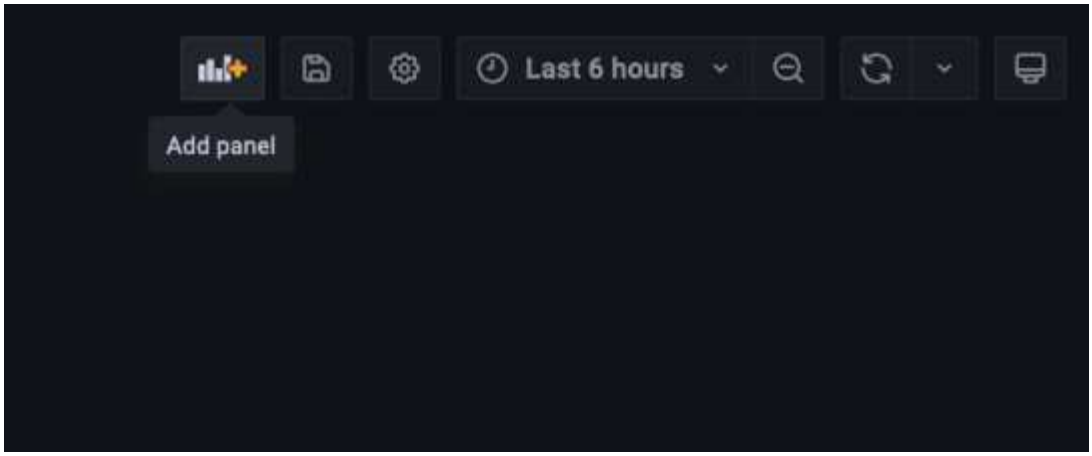


   c. From the Grid dashboard, lets select the "Storage Used - Object Metadata" panel. Click the little down arrow and the end of the panel title to drop down a menu. From this menu select "Inspect" and "Panel JSON".



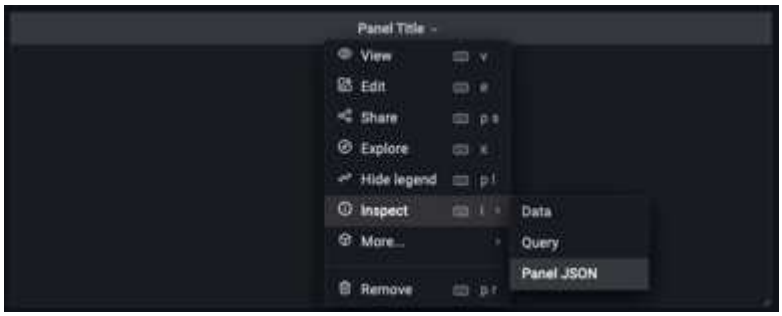
   d. Copy out the JSON code and close the window.

e. In our new dashboard, click on the icon to add a new panel.

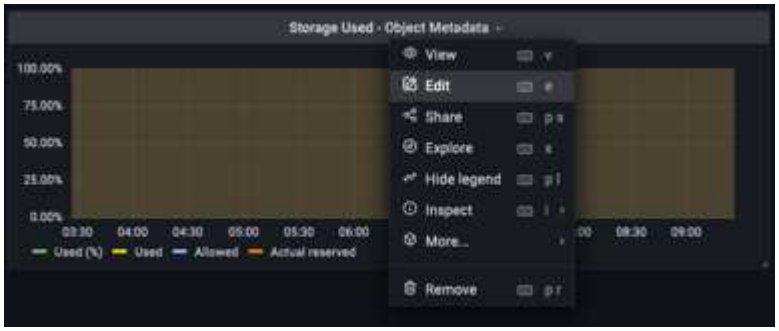


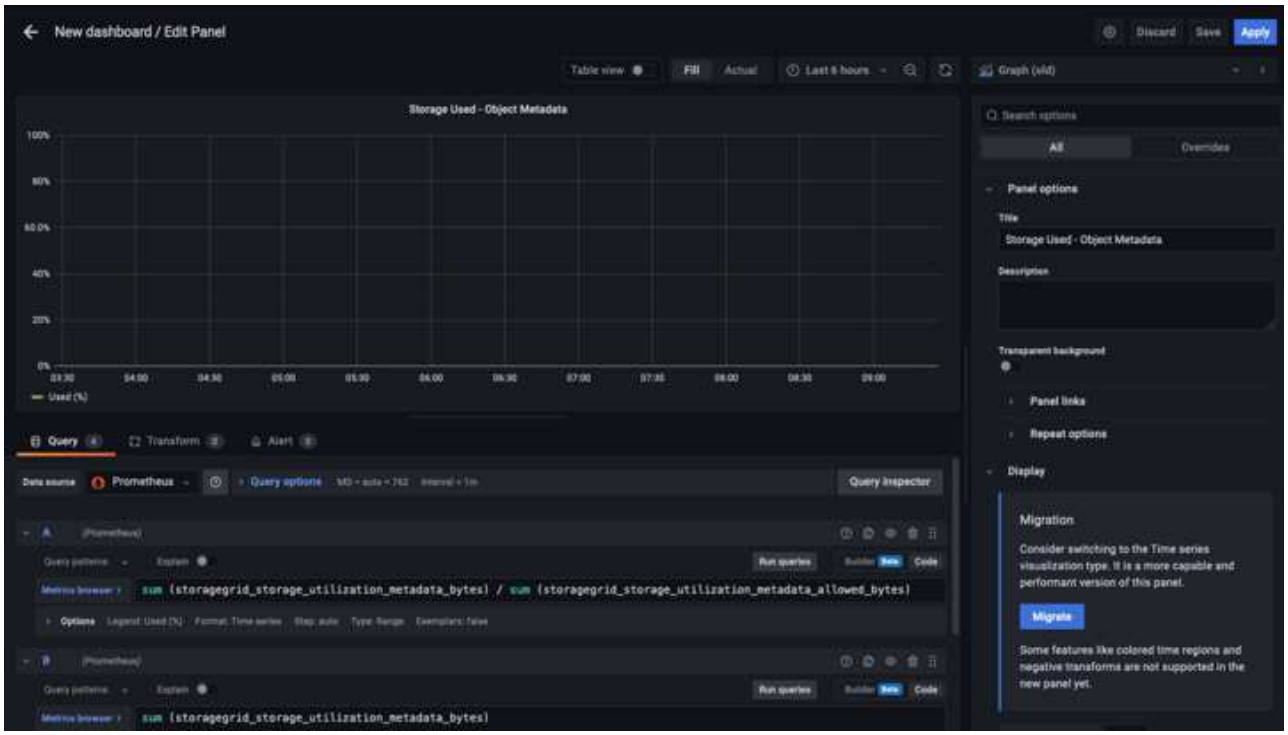
f. Apply the new panel without making any changes

g. Just like with the StorageGRID panel, inspect the JSON. Remove all JSON code and replace it with the copied code from the StorageGRID panel.



h. Edit the new panel, and on the right hand side you will see a Migration message with a "Migrate" button. Click the button and then click the "Apply" button.





8. Once you have all of the panels in place and configured as you like. Save the dashboard by clicking the disk icon in the upper right and give your dashboard a name.

Conclusion

Now we have a Prometheus server with customizable data retention and storage capacity. With this we can continue build out our own dashboards with the metrics that are most relevant to our operations. You can get more information on the Prometheus metrics collected in the StorageGRID documentation.

Use F5 DNS to globally load balance StorageGRID

By Steve Gorman (F5)

This technical report provides detailed instructions for configuring NetApp StorageGRID with F5 DNS services for global load balancing to deliver better data availability, greater data consistency, and optimize S3 transaction routing when your grid is distributed across multiple sites and/or HA groups.

Introduction

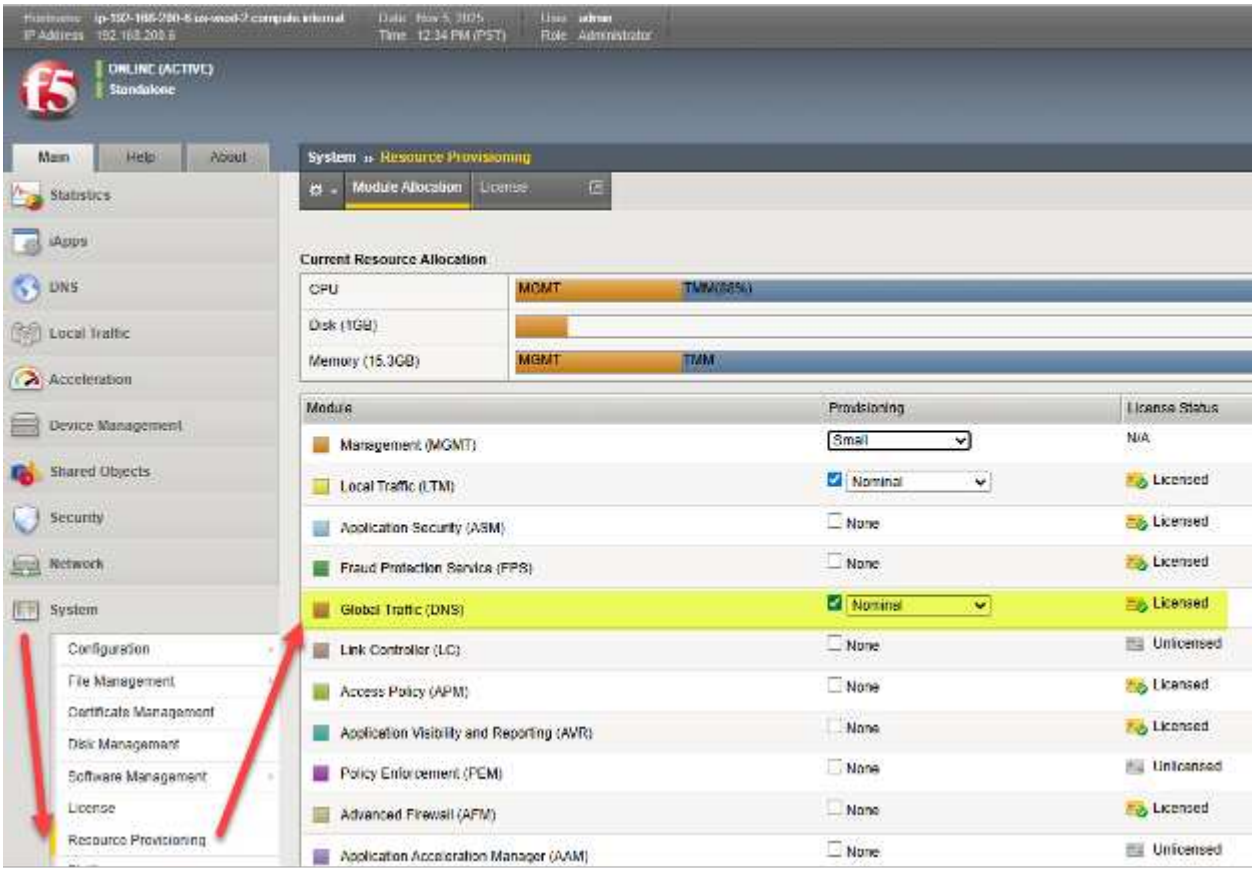
The F5 BIG-IP DNS solution formerly called BIG-IP GTM (Global Traffic Manager) and informally GSLB (Global Server Load Balancing) allows for seamless access across multiple active-active HA groups and active-active multi-site StorageGRID solutions to effectively be realized.

F5 BIG-IP multi-site StorageGRID configuration

Regardless of the number of StorageGRID sites to be supported, a minimum of two BIG-IP appliances, physical or virtual, must have the BIG-IP DNS module enabled and setup. The more DNS appliances, the further the degree of redundancy an enterprise will benefit from.

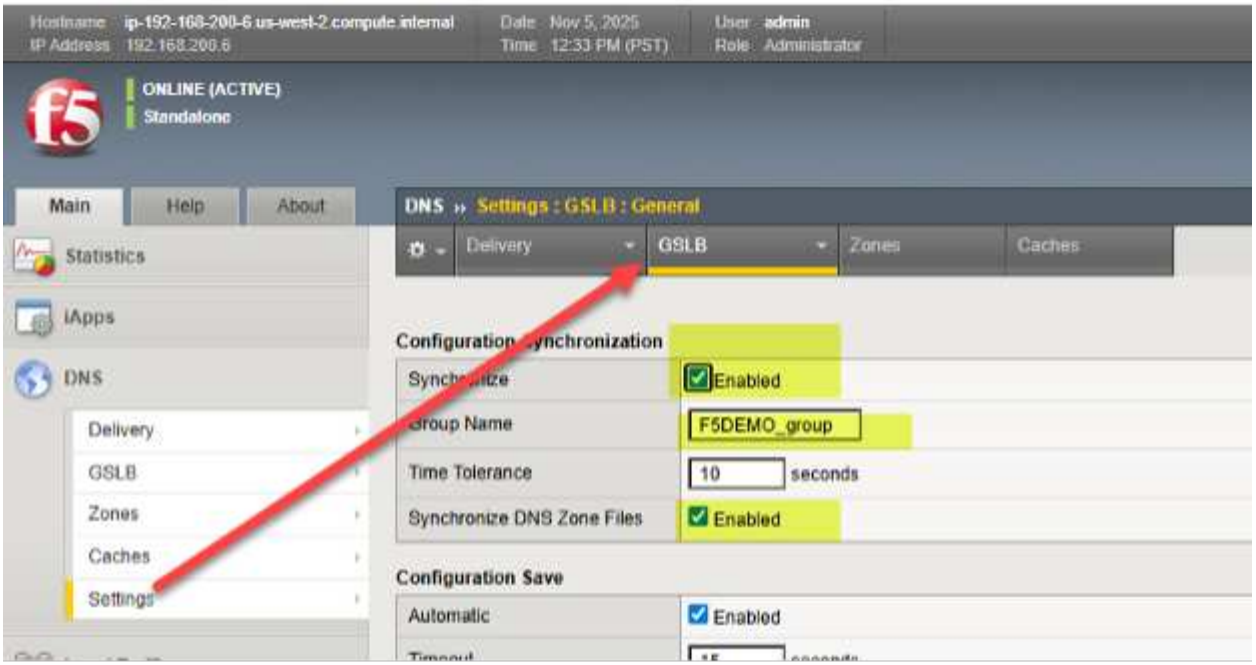
BIG-IP DNS - First Steps in Initial Set Up

Once the BIG-IP appliance has undergone at least initial provisioning, use a web browser to log into the TMUI (BIG-IP GUI) interface, and choose System → Resource Provisioning. As highlighted, ensure that the “Global Traffic (DNS)” module has a check mark and is shown to be licensed. Note, as in the image, it is common that “Local Traffic (LTM)” can be provisioned on the same appliance.

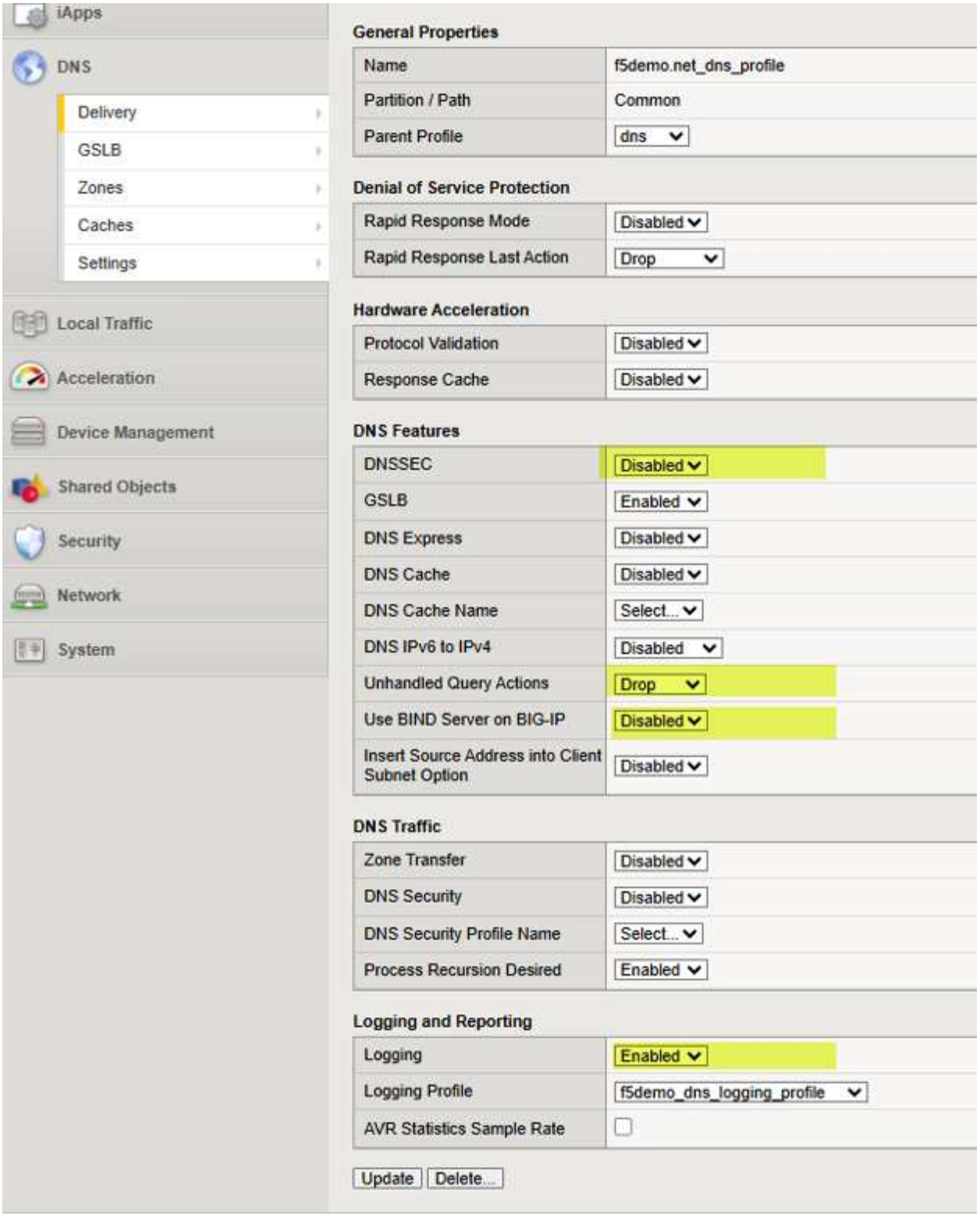


Configure DNS Protocol Foundational Elements

The first step towards global traffic management for StorageGRID sites is to choose the DNS tab, where virtually all the global traffic steering will be configured, and choose Settings→ GLSB. Enable the two synchronization options and choose a DNS group name that will be shared among participating BIG-IP appliances.



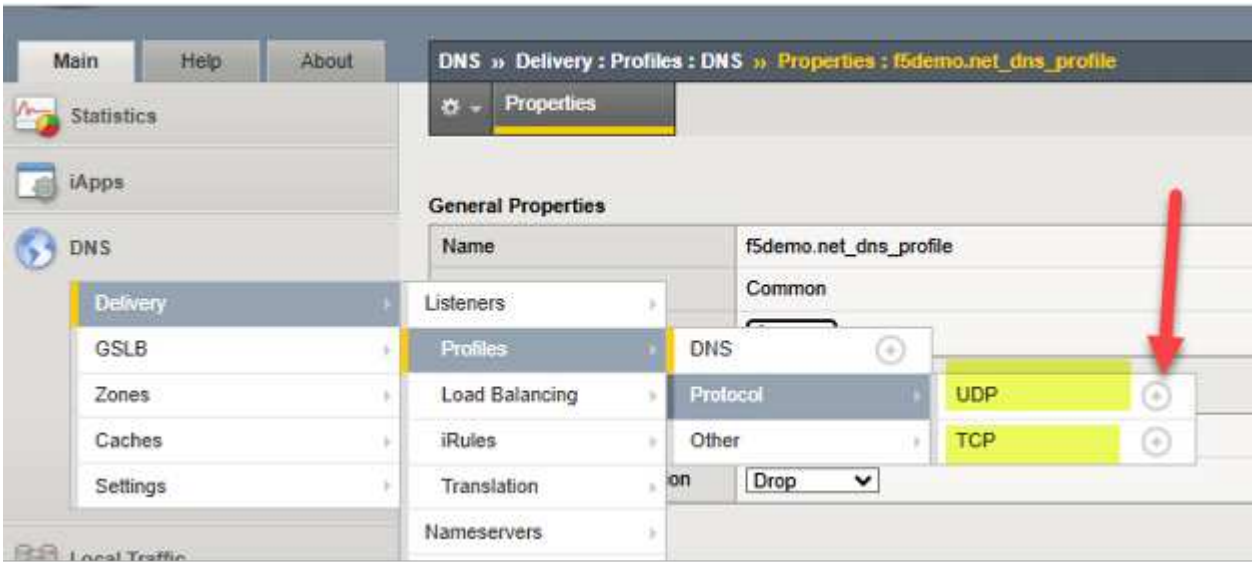
Next, navigate to DNS > Delivery > Profiles > DNS: Create and create a profile that will govern the DNS capabilities you wish to enable or disable. See the previous link for the DNS classroom guide if generation of specific DNS logs are of interest. Here is an example of a working DNS Profile, note the four highlights that represent settings that are important values. For awareness, each possible setting is explained at the following F5 KB (Knowledge Base) article here.



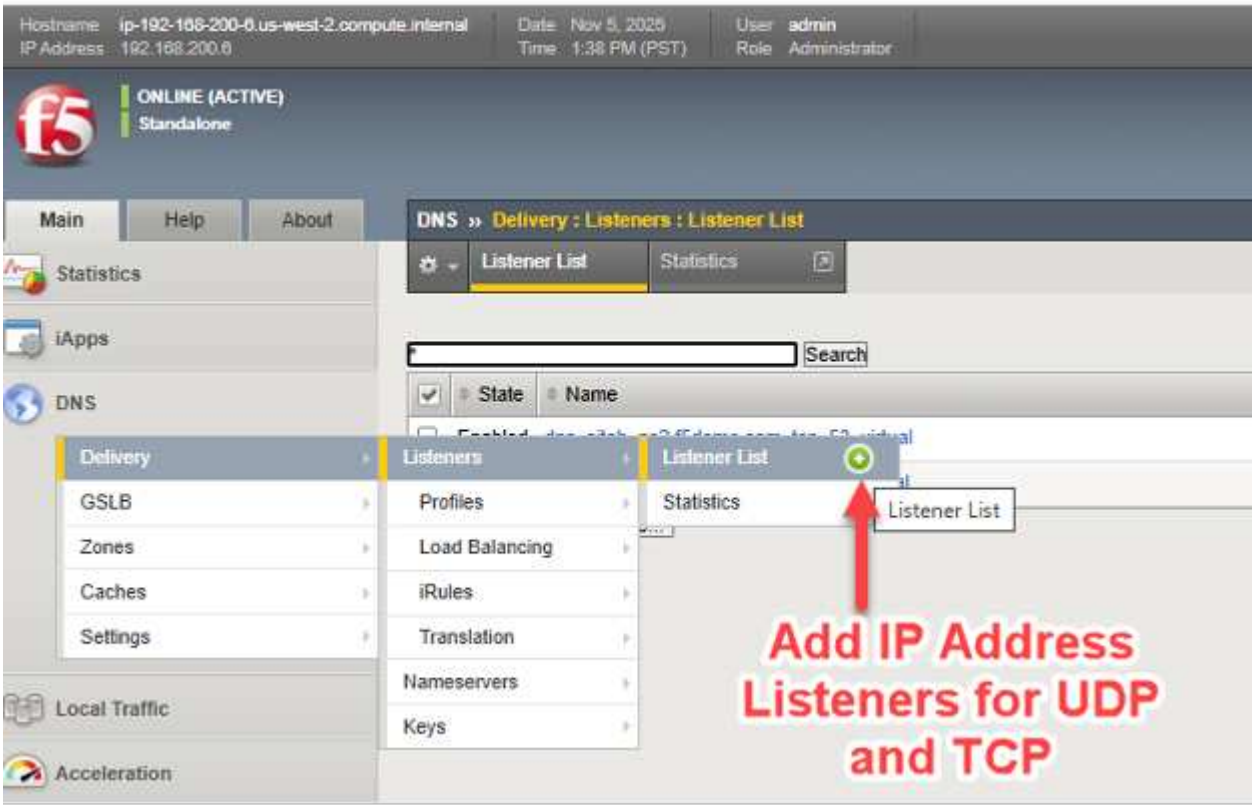
At this point we can adjust the characteristics of the UDP and TCP protocols, through created “profiles”, which can both carry DNS traffic involving BIG-IP. Simply create one new profile for UDP and TCP. Presuming DNS traffic will cross WAN links, a good practice is simply to inherit UDP and TCP characteristics known to perform well in WAN environments.
To add each simply click the “+” icon beside each protocol, and set the parent profile to the following:

UDP → use “parent” profile “udp_gtm_dns”

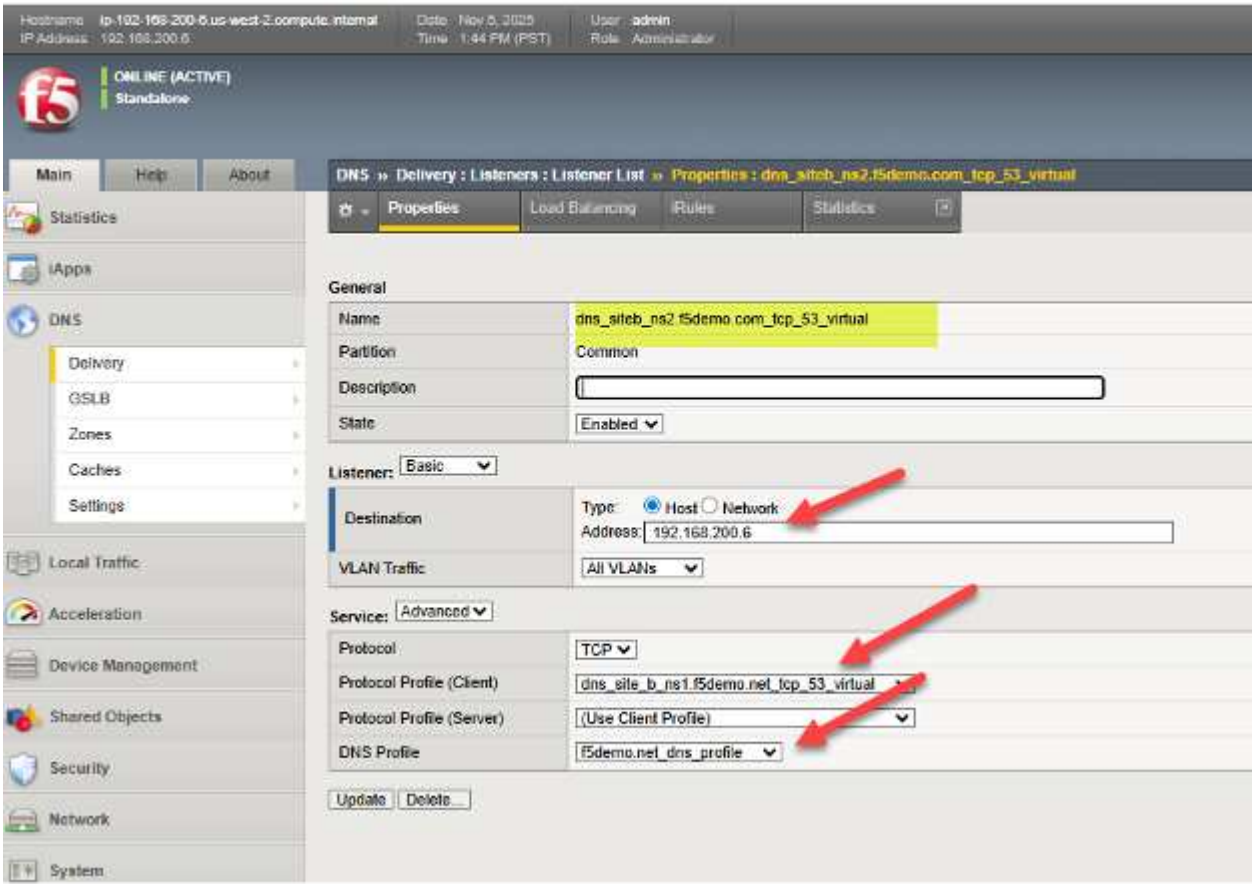
TCP → use “parent” profile “f5-tcp-wan”



Now, we simply need to assign an IP address for both UDP and TCP traffic involving the BIG-IP DNS. For those familiar with BIG-IP LTM, this is essentially the creation of DNS virtual servers, and virtual servers need “listener” IP addresses.
As in the screenshot, follow the arrows to create listener/virtual servers for DNS/UDP and DNS/TCP.



The following is one example from a live BIG-IP DNS, in it we see the TCP virtual server listener settings and can see how it ties together many of the previous steps. This includes referencing the DNS profile and protocol (TCP) profile, as well as configuring a valid IP address for DNS to use. As with all the objects one creates with BIG-IP, it is helpful to use a meaningful name which serves to self-identify what the object is, such as dns/siteb/TCP53 in the example name assigned.



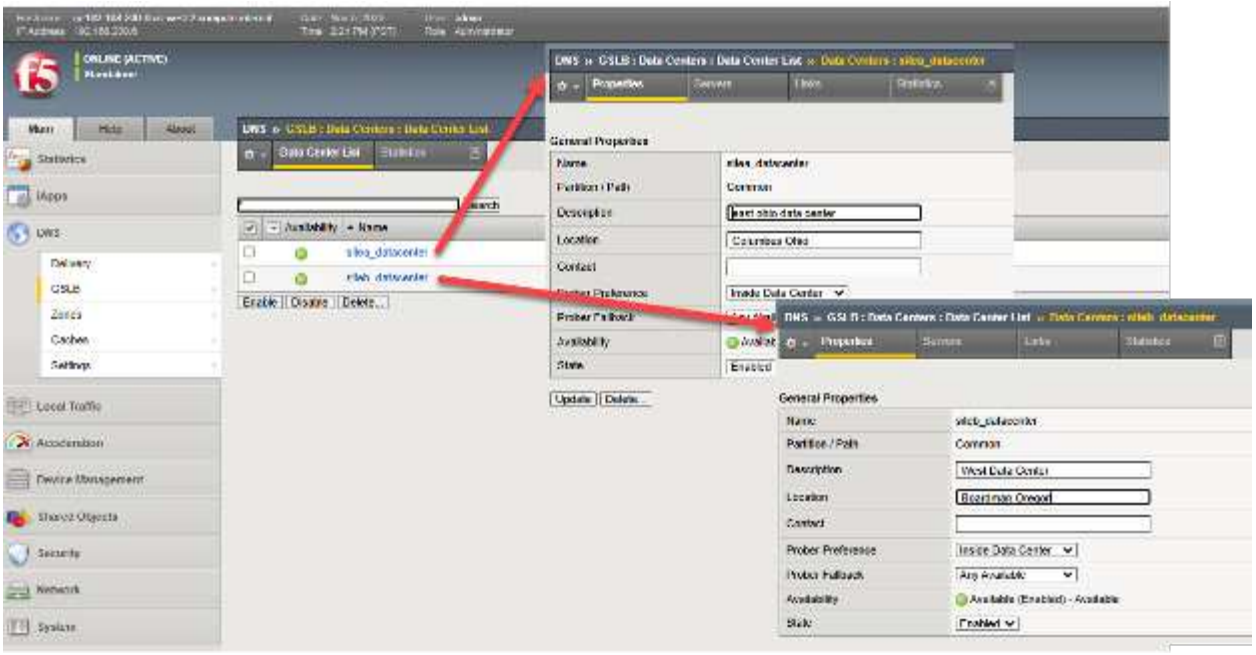
This concludes the preliminary, typically “one time”, setup steps of a BIG-IP appliance with the DNS module enabled. At this point we are ready to transition to the specifics of setting up a global traffic management solution with our appliances, which will of course be linked to the characteristics of the StorageGRID sites.

Setting up Data Center Sites and Establishing Inter BIG-IP Communications in Four Steps

Step one: Create Data Centers

Each site which will house clusters of nodes to be locally load balanced by BIG-IP LTM, should be entered into BIG-IP DNS. This needs to be done on only one BIG-IP DNS, as we are creating a DNS synchronized group to support traffic management, as such this configuration will be shared among DNS members of the group.

Through the TMUI GUI, select DNS > GSLB > Data Centers > Data Center List and create an entry for each of the StorageGRID sites. If using a network setup aligned with Figure 1, DNS appliance located in other non-StorageGRID sites, add Data Centers for these sites in addition to storage sites. In this example sites a and b are created in Ohio and Oregon, the BIG-IPs are dual DNS and LTM appliances.



Step two: Create Servers (List of All BIG-IP Appliances in Solution)

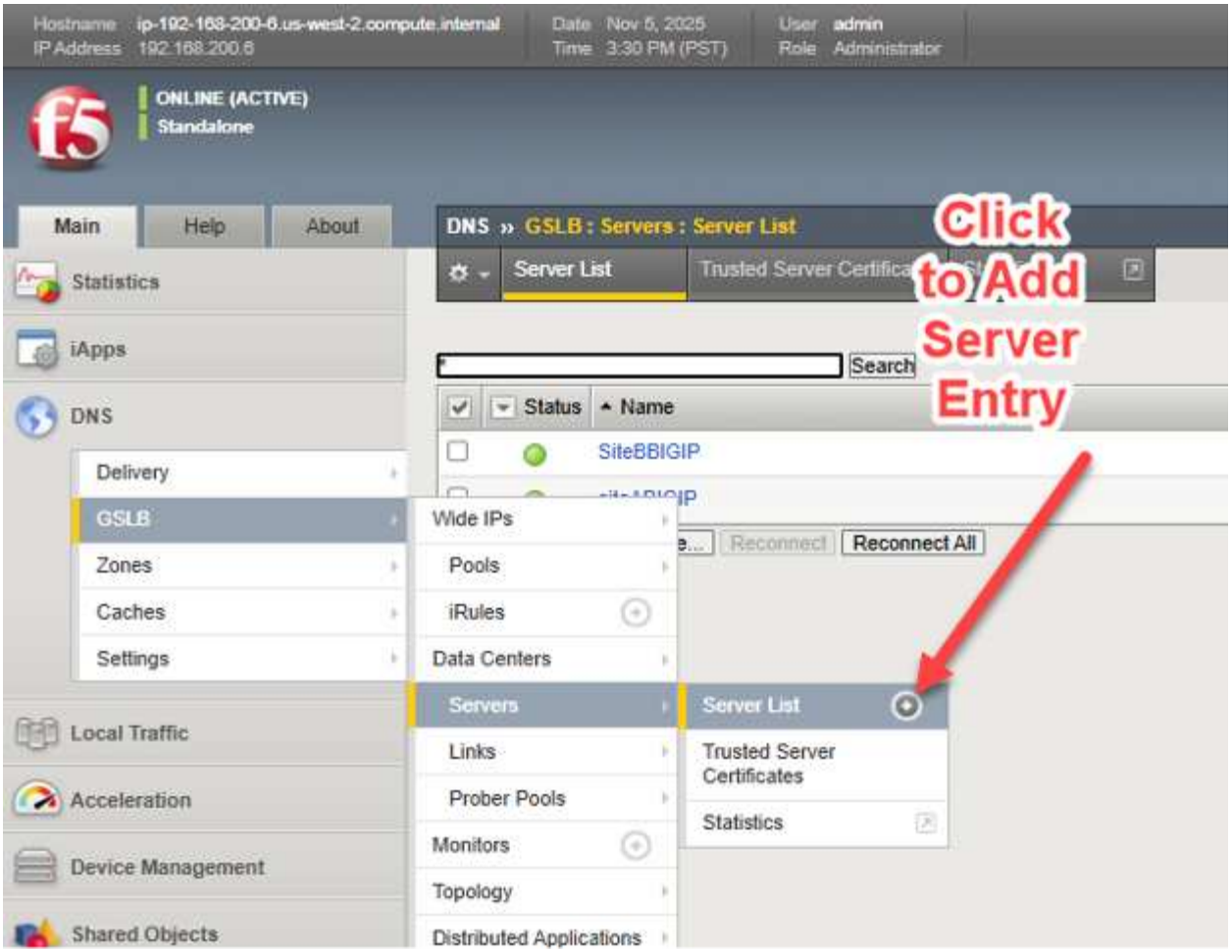
We are now ready to connect the individual StorageGRID site clusters to the BIG-IP DNS setup. Recall, the BIG-IP appliance at each site will do the actual load balancing of S3 traffic, through the configuration of virtual servers that tie a “front-end” reachable IP address/port to a set of back end “pool” of Storage Node appliances, using “back-end” IP addresses/ports.

Should, as one example, all Storage Nodes in a pool be taken offline administratively, perhaps for a site decommissioning, or unexpectedly through real-time failed health checks, traffic will be directed to other sites through altering DNS query responses.

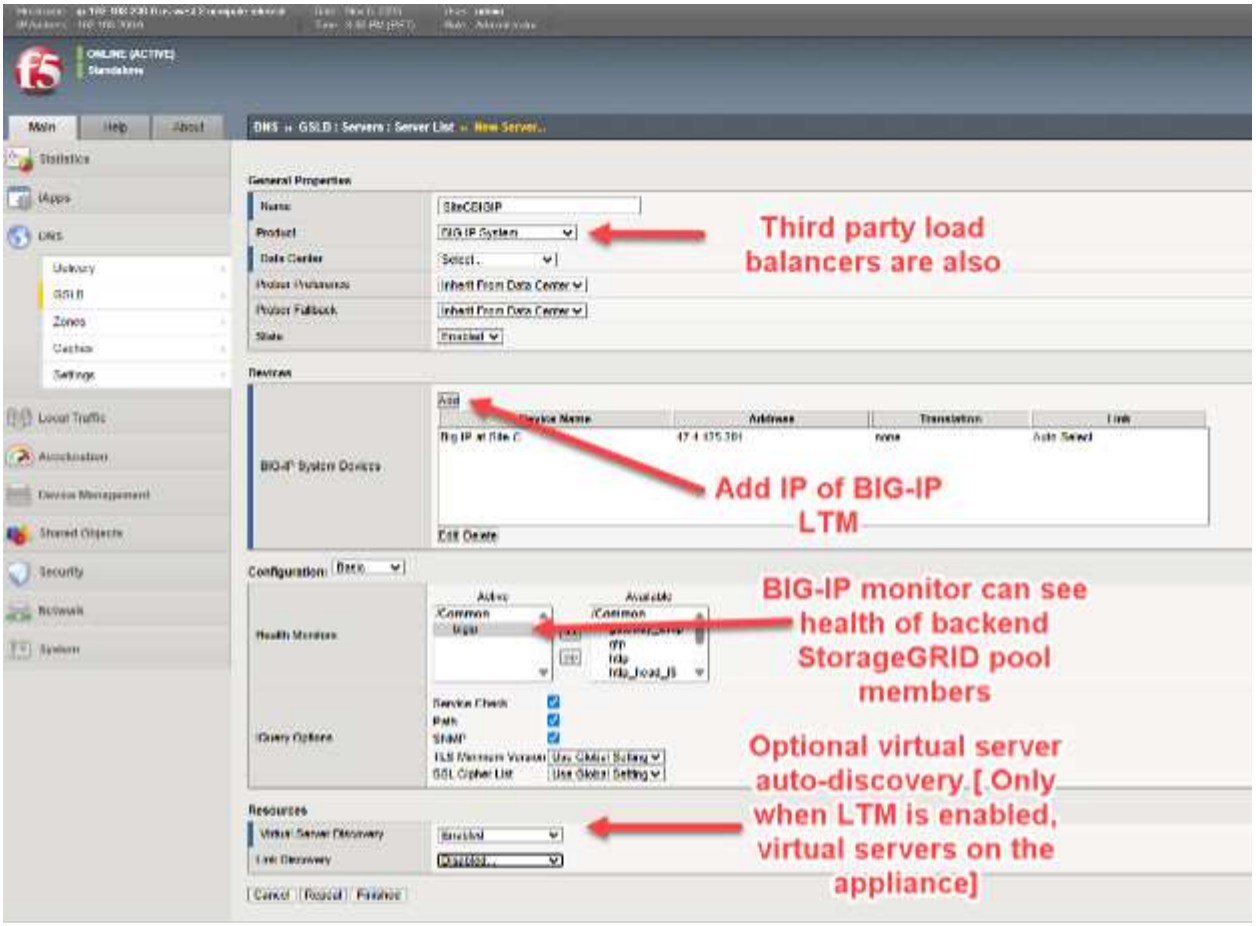
To tie the StorageGrid sites, specifically the local virtual servers, into the BIG-IP DNS configuration on each appliance, the setup need only be done once. The entire group of BIG-IP DNS appliances will have their setups synchronized, in an upcoming step.

In simple terms, we will create a list, referred to as a server list, of all of our BIG-IP appliances, whether licensed for DNS, LTM or both DNS and LTM. This master list will be sync’d with all BIG-IP DNS appliances upon completion of the list.

On one BIG-IP DNS licensed appliance, choose DNS > GSLB > Servers > Server List and choose the add button (+).



The four key elements when adding each BIG-IP include:
* Selecting BIG-IP from the product pull down, other load balancers are possible but generally lack the real time visibility responsiveness when backend node health deteriorates at each site.
* Add the IP address of the BIG-IP DNS appliance. likely, the first time adding a BIG-IP DNS appliance, the address will be the current GUI-accessed appliance, future appliances will the other appliances in the solution.
* Choose a health monitor, always use “BIG-IP” when the load balancer being added is BIG-IP appliance, for back-end StorageGRID node health consideration.
* Optionally, request virtual server automatic discovery if the appliance is a dual DNS/LTM appliance.

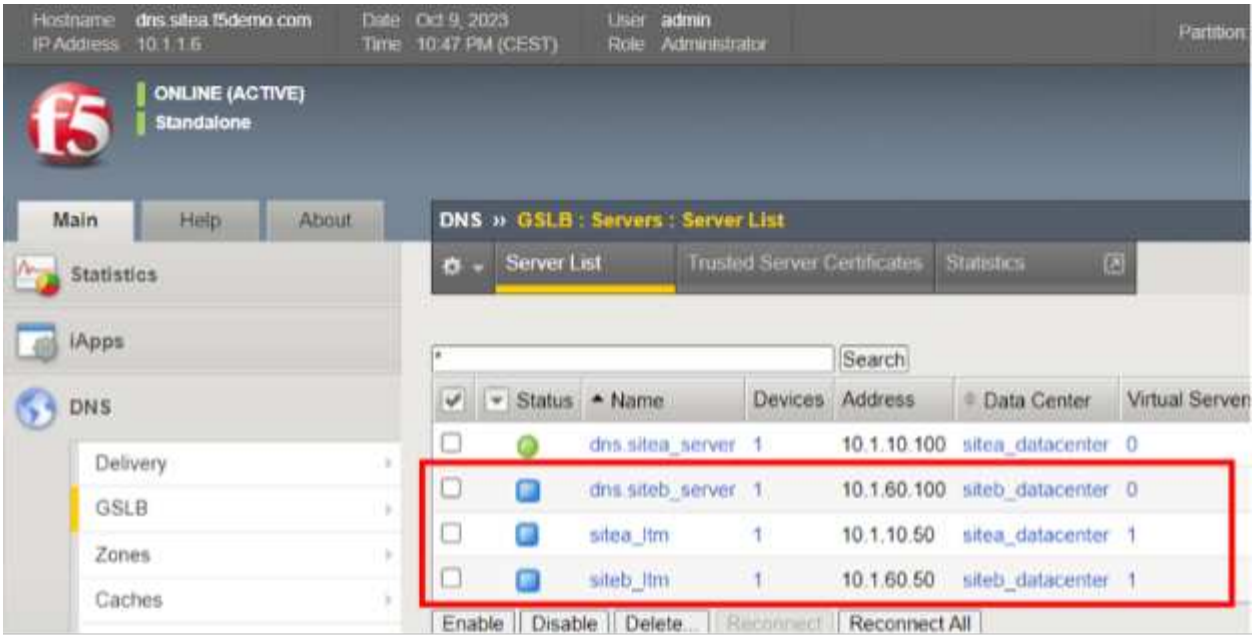


In some situations, such as transient networking problems or firewall ACL rules between network locations, when adding a remote appliance at this stage, the virtual server discovery may not show entries for remote appliances with LTM configured. In such cases, after adding the new appliance (“server”), one can manually add the virtual servers as indicated below. If adding a BIG-IP DNS-only appliance, there will be no virtual servers to be discovered or added to that device.

We need to add these server entries for each appliance in our solution at all sites, including BIG-IP DNS appliances, BIG-IP LTM appliances, and any appliances serving the dual roles of both DNS and LTM units.

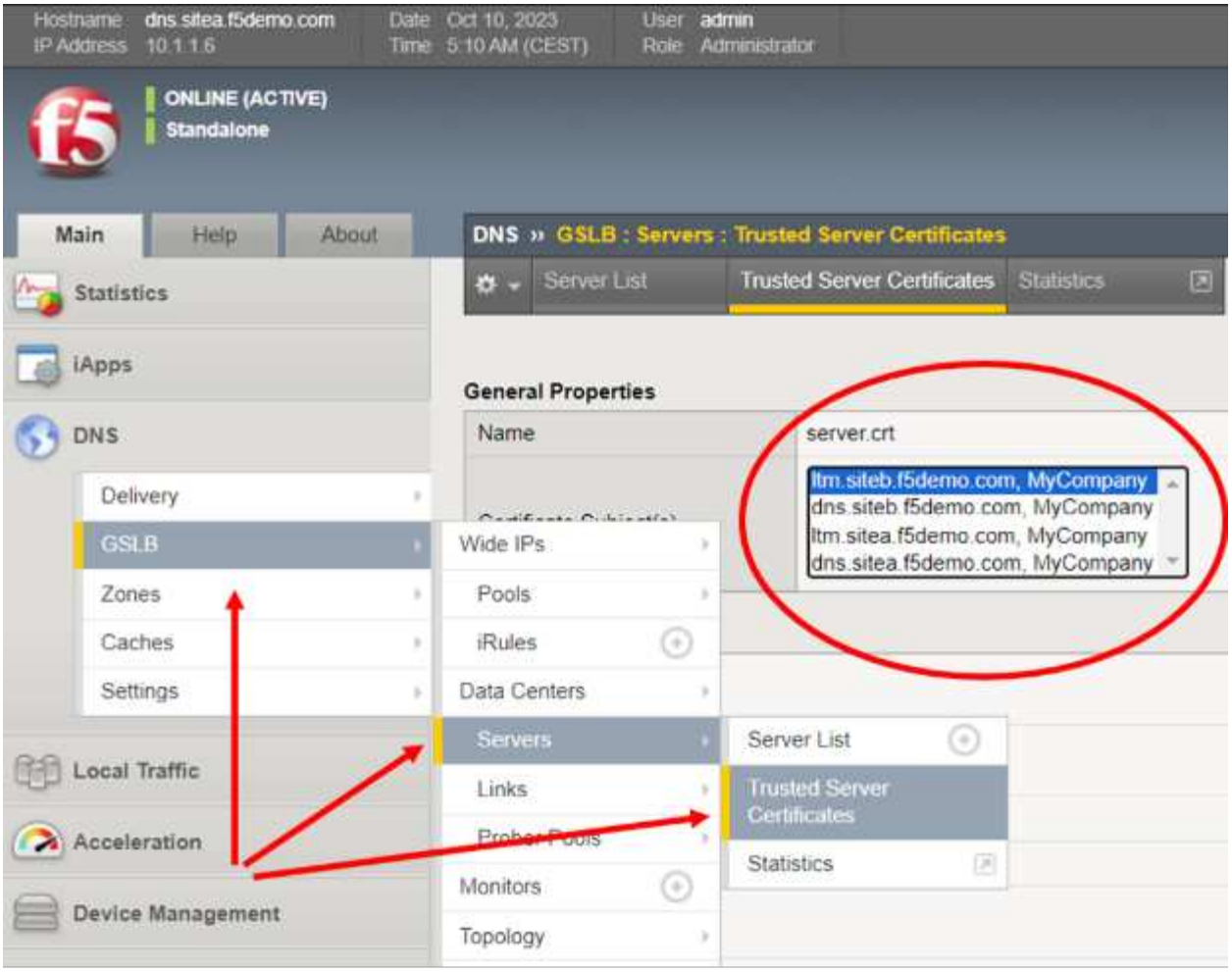
Step three: Establish Trust Between all BIG-IP Appliances

In the following example, four appliances have been added as servers, they are spread across two sites. Note each site has a dedicated BIG-IP DNS and BIG-IP LTM. However, all of the appliances, other than the one currently logged into, are showing blue icons in the “Status” column. This means a trust relationship has not yet been established with the other BIG-IP appliances.



To add trust, SSH into the BIG-IP where the configuration details have just been entered via the GUI, use the “root” account to access the BIG-IP command line interface.
Issue the following singe command at the prompt: *bigip_add*

The "bigip_add" command pulls the management certificate from the destination BIGIP devices for use during the encrypted “iQuery” channel setup between GSLB servers in the cluster. iQuery, by default, runs using TCP port 4353 and is the hearbeat that allows BOG-IP DNS members to stay in a synchronized state. It makes use of xml and gzip in the encrypted channel. When running "bigip_add" without any options, the command will be run against all BIGIP devices in the GSLB Server list using the current username to connect to the endpoints. As a quick check of success, simply return to the BIG-IP GUI and confirm all servers now have certificates listed in the displayed pull-down menu.



Step four: Synchronize all BIG-IP DNS Appliances to the DNS Group

The final step will allow all BIG-IP DNS appliances to be fully configured by simply using the TMUI GUI of one single unit. In a sample case, where there are two StorageGRID sites, this means now using SSH to reach the command line of the **other** site's BIG-IP DNS.
After connecting as root, and ensuring that firewall policies/ACLs allow the two BIG-IP DNS devices to talk on TCP ports 22 (SSH), 443 (HTTPS) and 4354 (F5 iQuery protocol), issue this one command at the prompt:
gtm_add <IP address of first site BIG-IP DNS, where all of the GUI steps were previously done>

At this point all further DNS configuration work can be performed on any BIG-IP DNS appliance that has been added to the group. The above command, gtm_add, need not be applied on appliance members that are LTM only. Only appliances supporting DNS require this command to become part of the synchronized DNS group.

Setting up Data Center Sites and Establishing Inter BIG-IP Communications

At this point, all the steps to create the underlying, healthy BIG-IP DNS appliance group is complete. We can now get on with creating names, FQDNs, that point towards our distributed web/S3 services exposed at each StorageGRID datacenter.

These names are referred to as "Wide IPs", or WIPs for short, and they are normal DNS FQDNs with DNS A resource records. However, rather than pointing at a server like a traditional A resource record, they internally point at pools of BIG-IP virtual servers. Each pool, individually, can be made up of a set of one or more virtual servers. An S3 client requesting an IP address to name resolution will receive the address of the S3 virtual server at the optimal, policy-selected StorageGRID site.

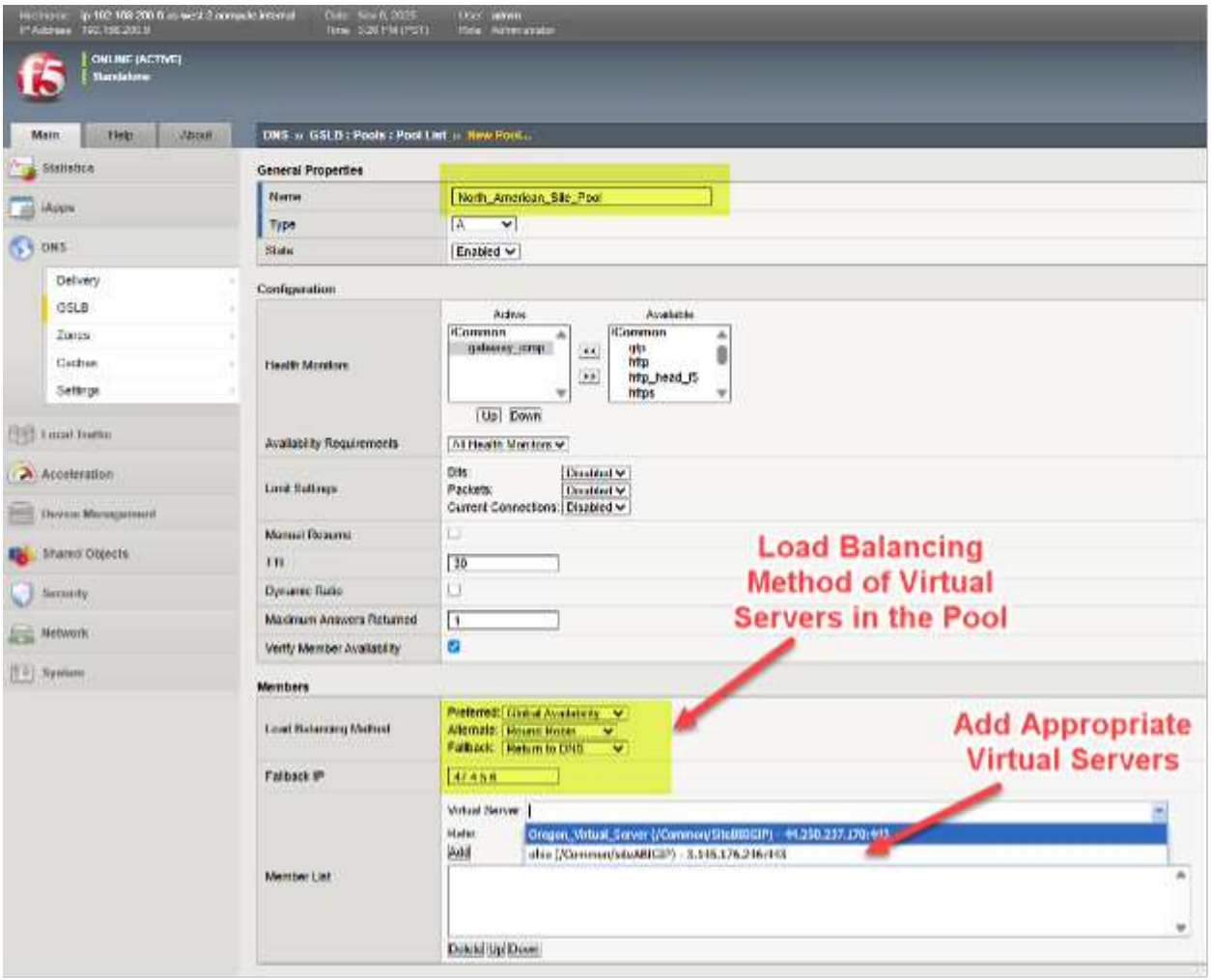
Wide IPs, Pools and Virtual Servers in a Nutshell

To give a simple, fictitious example, a WIP for the name **storage.quantumvault.com** might see the BIG-IP DNS solution linked with two pools of potential virtual servers. The first pool might be made up of 4 sites in North America; the second pool might consist of 3 sites in Europe.

The selected pool might be arrived at from a range of policy decisions, perhaps a simple ratio of 5:1 could be used to have the bulk of traffic directed to North American StorageGRID sites. More probable perhaps, a topology-based choice where the pool is chosen where, for instance, all European sourced S3 traffic is directed to European sites, and the remainder of world S3 traffic is directed to North American data centers.

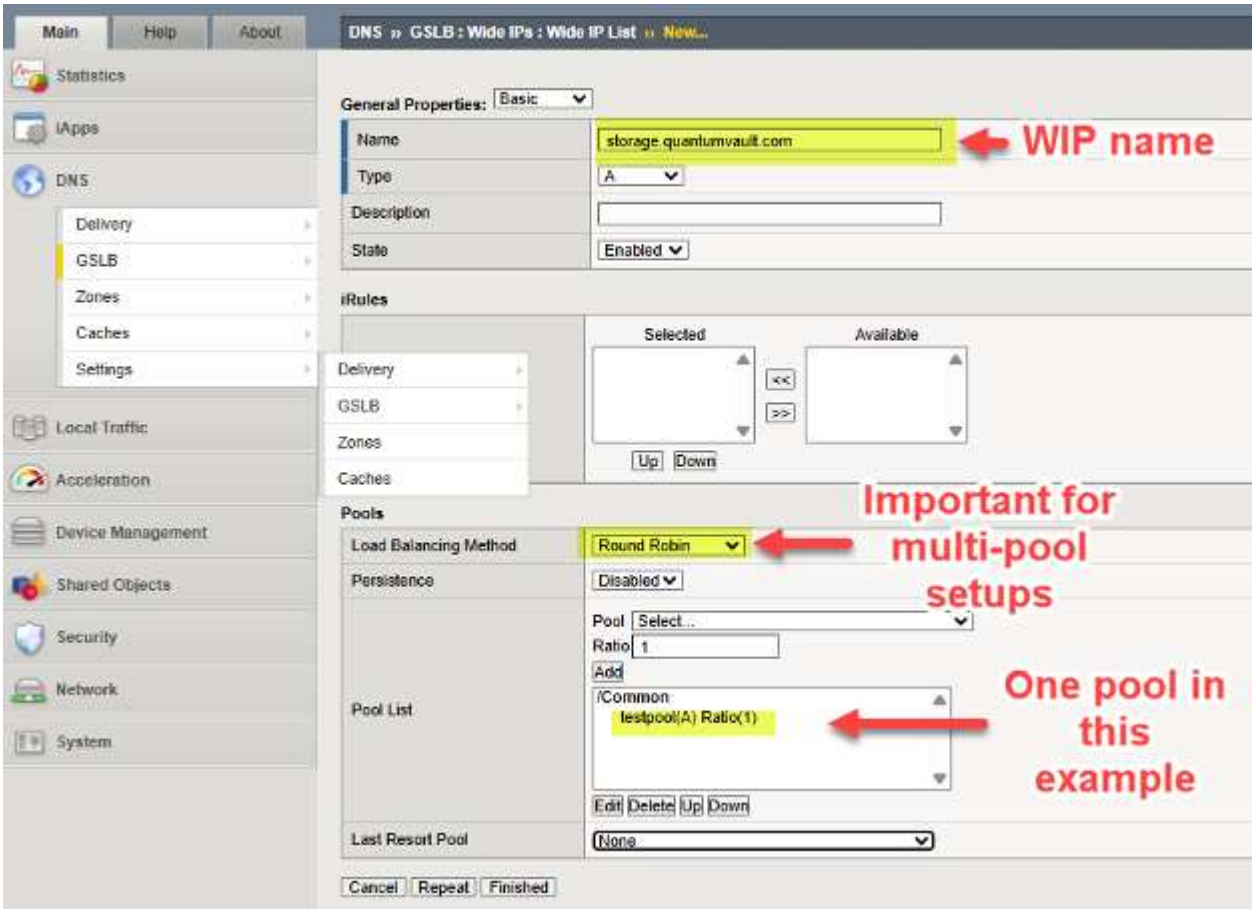
Once a pool is arrived at by BIG-IP DNS, let us assume the North American pool was selected, the actual DNS A Resource Record returned to resolve storage.quantumvault.com can be any one of the 4 virtual servers supported by BIG-IP LTM in any of the 4 North American sites. Again, which is chosen is policy driven, simple “static” approaches such as Round-Robin exist, whereas more advanced “dynamic” selections such as performance probes to measure each sites latency from local DNS resolvers is maintained and used as the criteria for site selection.

To set a Pool of virtual servers on a BIG-IP DNS, follow the menu path **DNS > GSLB > Pools > Pool List > Add (+)**.
In this example, we can see various North American virtual servers are added to a pool and the preferred approach to load balancing, when this pool is selected, is chosen in a tiered fashion.



We add the WIP (Wide IP), the name of our service which will be resolved by DNS, to a deployment by following the DNS > GSLB > Wide IPs > Wide IP List > Create (+). In the following example, we provide an

example WIP for an S3-enabled storage service.



Adjust DNS to Support Global Traffic Management

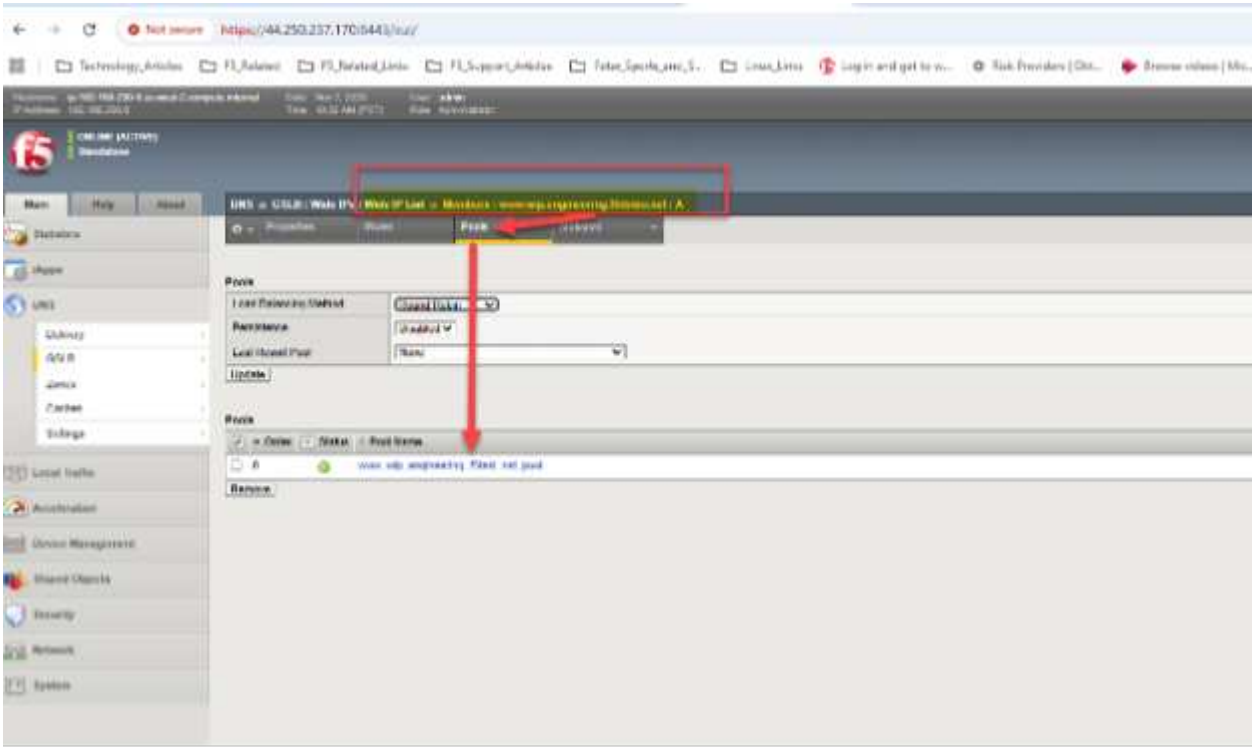
At this point all of our underlying BIG-IP appliances are ready to perform GSLB (global server load balancing). We simply need to adjust and assign the names used for S3 traffic flows to leverage the solution.
The general approach is to delegate part of an existing DNS domain of an enterprise to the control of BIG-IP DNS. This is to say to “carve” a section of the name space, a sub-domain, and delegate control of this sub-domain to the BIG-IP DNS appliances. Technically, this is done by ensuring the BIG-IP DNS appliances have A DNS resource records (RRs) in the enterprise DNS and then making these names/addresses Name Server (NS) DNS resource records for the delegated domain.

There are various ways enterprises maintain DNS today, one method is a fully hosted solution. An example of this would be operating and managing DNS through Windows Server 2025. An alternative approach can be for an enterprise to leverage cloud-DNS providers like AWS Route53 or Squarespace.

Here is a fictious example for illustration purposes. We have StorageGRID supporting object reads and writes via the S3 protocol with an existing domain managed by AWS Route53, the existing example domain is f5demo.net.

We would like to assign the sub-domain engineering.f5demo.net to the BIG-IP DNS appliances for global traffic management. To do this, we create a new NS (name server) resource record for engineering.f5demo.net and point that to the list of BIG-IP DNS appliance names. In our example, we have two BIG-IP DNS appliances, and as such we create two A resource records for them.

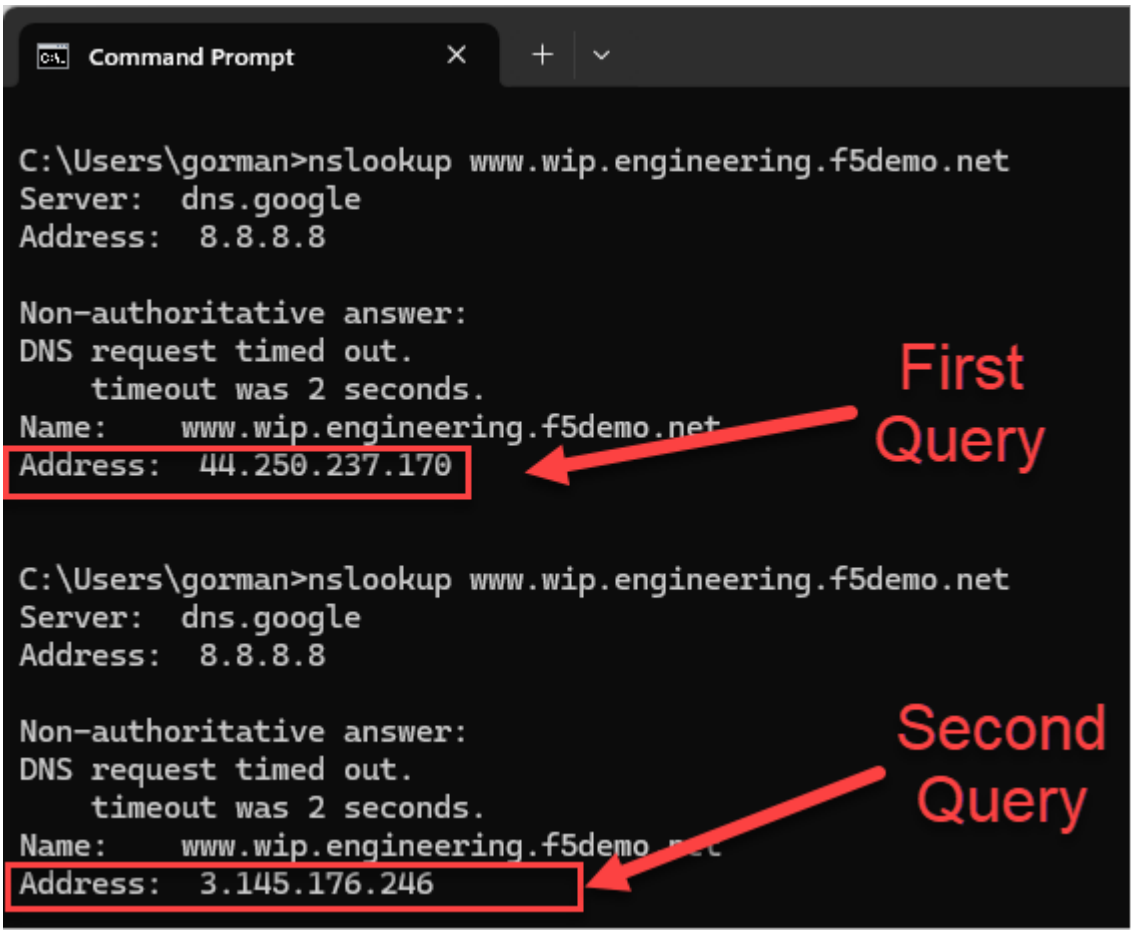
We now, as an example, will set up a Wide IP (WIP) in our BIG-IP DNS, since DNS uses group synchronization, we only need to adjust using the GUI of one appliance. Within the BIG-IP DNS GUI, go to **DNS > GSLB > Wide IPs > Wide IP List (+)**. Recall, in a traditional DNS FQDN setup one would be entering one or more IPv4 addresses, in our case we simply point at one or more pools of StorageGRID virtual servers.



In our example, we have generic web HTTPS servers located in both Ohio and Oregon sites. With a simple “round robin” approach, we should be able to see the global DNS respond to queries for the A resource record mappings for *www.wip.engineering.f5demo.net* with both virtual server IPs.

A simple test can be done with web browsers or, in the case of S3 using StorageGRID, perhaps graphical tools like S3Browser. Each DNS query will see the next data center site in the pool used as the target for ensuing traffic, due to our choice of Round Robin within the pool.

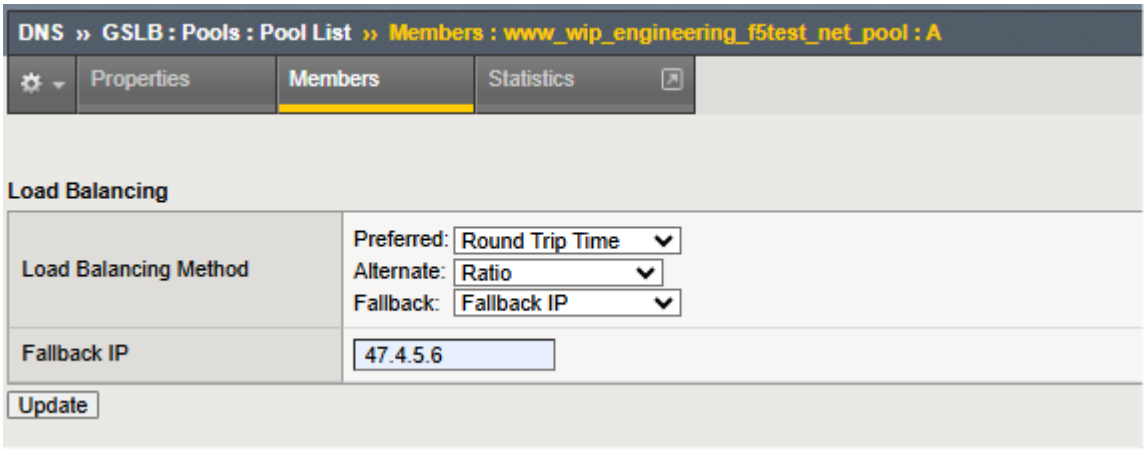
In our example setup, we can use dig or nslookup to quickly generate a series of two DNS queries and ensure BIG-IP DNS is indeed doing a round robin load balancing, resulting in both sites receiving traffic over time.



Suggested exploration for more advanced techniques

One of many possible approaches, would use “Global Availability” mode as opposed to the simple “Round Robin” example given above. With Global Availability the sequenced order of pools, or virtual servers within just a single pool, can have traffic directed to it. In this way, all S3 traffic could by default, be directed towards, say, a New York City site.

If health checks indicate an issue with StorageGRID node availability at this site, traffic could at that point be directed to St. Louis. Should St. Louis encounter health concerns, a site in Frankfurt could in turn begin to receive S3 read or write transactions. Thus, global availability is one approach to S3 StorageGRID overall solution resiliency.
Another approach is to mix and match load balancing approaches, where a tiered approach is used.



In this example, a “dynamic” option is the first load balancing choice for the sites in the configured pool. In the example shown, an on-going measurement approach using active probing of local DNS resolver performance is maintained and the catalyst for site selection. Should this approach be unavailable, the individual sites can be selected by the ratio assigned to each. With ratio, larger, higher-bandwidth StorageGRID sites can receive more S3 transactions than smaller sites.
Finally, as perhaps a disaster recovery scenario, should all sites in the pool become unhealthy, the specified fallback IP is used as the site of last resort.
One of the more interesting load balancing methods of BIG-IP DNS is “Topology” whereby the incoming source of DNS queries, the S3 user’s local DNS resolver, is observed and using Internet topology information the seemingly “closest” site is selected from the pool.

Lastly, if sites span the entire globe, it may be worth considering using the dynamic “probe” technology discussed in detail in the F5 BIG-IP DNS manual. With probes, frequent sources of DNS queries can be monitored, take for example a business-to-business partner whose traffic generally uses the same local DNS resolver. BIG-IP DNS probes can be launched from the BIG-IP LTM in each site around the globe, to determine generally which potential site would likely offer the lowest latency for S3 transactions. As such, traffic from Asian might be better served by Asian StorageGRID sites than sites located in North American or Europe.

Conclusion

The integration of F5 BIG-IP with NetApp StorageGRID addresses technical challenges related to data availability and consistency across multiple sites and optimizing S3 transaction routing. Deploying this solution enhances storage resilience, performance, and reliability, making it ideal for enterprises seeking a robust, scalable, and flexible storage infrastructure.

To learn more, the official F5 documentation for BIG-IP DNS can be found at this link.
A guided classroom style guide which provides step-by-step instructions on an example setup can also be found here.

Datadog SNMP configuration

By Aron Klein

Configure Datadog to collect StorageGRID snmp metrics and traps.

Configure Datadog

Datadog is a monitoring solution providing metrics, visualizations, and alerting. The following configuration was implemented with linux agent version 7.43.1 on an Ubuntu 22.04.1 host deployed local to the StorageGRID system.

Datadog Profile and Trap files Generated from StorageGRID MIB file

Datadog provides a method for converting product MIB files into datadog reference files required to map the SNMP messages.

This StorageGRID yaml file for Datadog Trap resolution mapping generated following the instruction found here.
Place this file in /etc/datadog-agent/conf.d/snmp.d/traps_db/ +

- Download the trap yaml file +
  - **md5 checksum** 42e27e4210719945a46172b98c379517 +
  - **sha256 checksum** d0fe5c8e6ca3c902d054f854b70a85f928cba8b7c76391d356f05d2cf73b6887 +

This StorageGRID profile yaml file for Datadog metrics mapping generated following the instruction found here.
Place this file in /etc/datadog-agent/conf.d/snmp.d/profiles/ +

- Download the profile yaml file +
  - **md5 checksum** 72bb7784f4801adda4e0c3ea77df19aa +
  - **sha256 checksum** b6b7fadd33063422a8bb8e39b3ead8ab38349ee0229926eadc8585f0087b8cee +

SNMP Datadog configuration for Metrics

Configuring SNMP for metrics can be managed in two ways. You can configure for auto-discovery by providing a network address range containing the StorageGRID system(s), or define the IP's of the individual devices. The configuration location is different based on the decision made. Auto-discovery is defined in the datadog agent yaml file. Explicit device definitions are configured in the snmp configuration yaml file. Below are examples of each for the same StorageGRID system.

Auto-discovery

configuration located in /etc/datadog-agent/datadog.yaml

```
listeners:
  - name: snmp
snmp_listener:
  workers: 100  # number of workers used to discover devices concurrently
  discovery_interval: 3600  # interval between each autodiscovery in
seconds
  loader: core  # use core check implementation of SNMP integration.
recommended
  use_device_id_as_hostname: true  # recommended
  configs:
    - network_address: 10.0.0.0/24  # CIDR subnet
      snmp_version: 2
      port: 161
      community_string: 'st0r@gegrid'  # enclose with single quote
      profile: netapp-storagegrid
```

Individual devices

/etc/datadog-agent/conf.d/snmp.d/conf.yaml

```
init_config:
  loader: core  # use core check implementation of SNMP integration.
recommended
  use_device_id_as_hostname: true  # recommended
instances:
- ip_address: '10.0.0.1'
  profile: netapp-storagegrid
  community_string: 'st0r@gegrid'  # enclose with single quote
- ip_address: '10.0.0.2'
  profile: netapp-storagegrid
  community_string: 'st0r@gegrid'
- ip_address: '10.0.0.3'
  profile: netapp-storagegrid
 community_string: 'st0r@gegrid'
- ip_address: '10.0.0.4'
  profile: netapp-storagegrid
  community_string: 'st0r@gegrid'
```

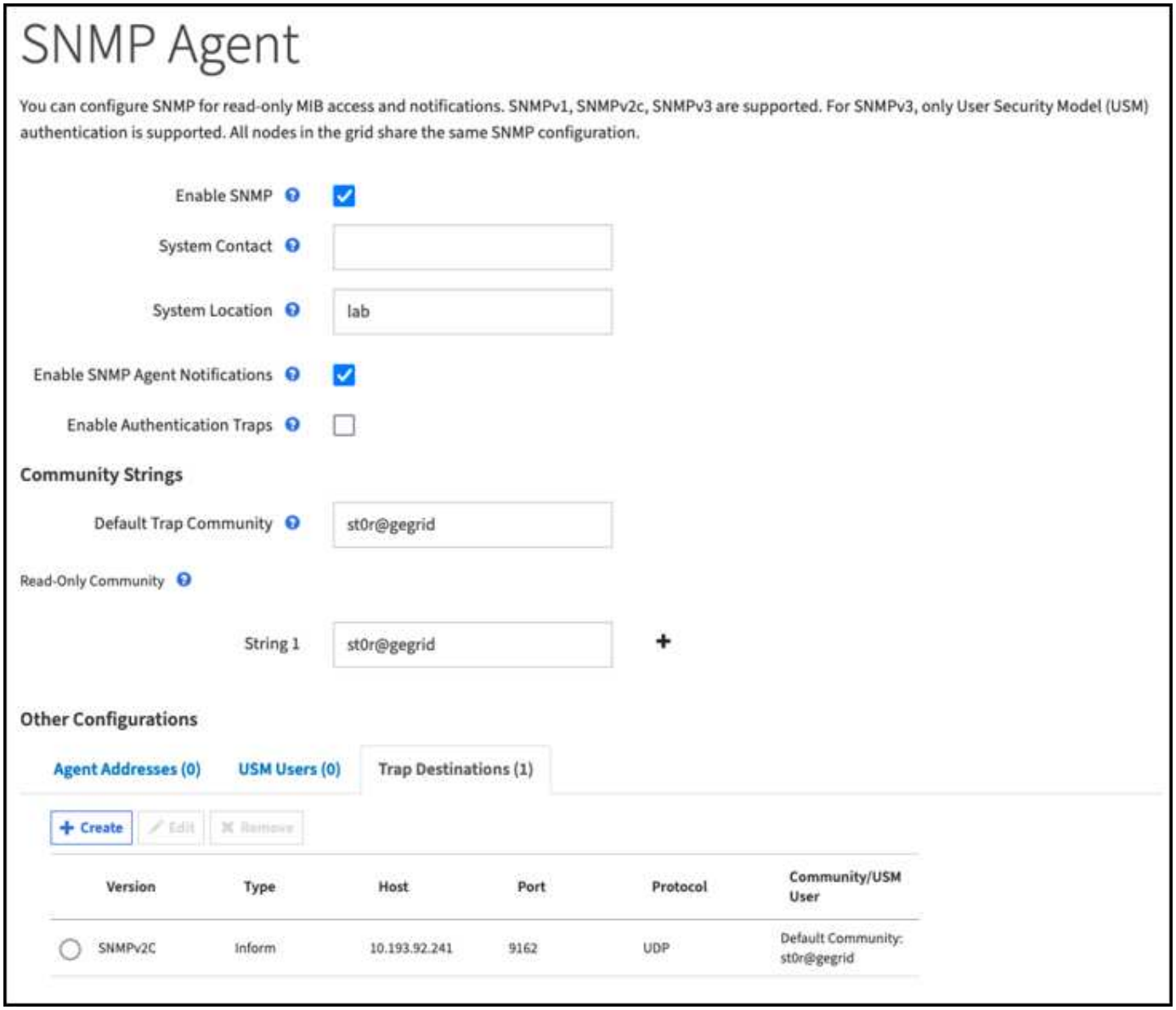
SNMP configuration for traps

The configuration for SNMP traps is defined in the datadog configuration yaml file /etc/datadog-agent/datadog.yaml

```
network_devices:
  namespace:  # optional, defaults to "default".
  snmp_traps:
    enabled: true
    port: 9162 # on which ports to listen for traps
    community_strings: # which community strings to allow for v2 traps
      - st0r@gegrid
```

Example StorageGRID SNMP configuration

The SNMP agent in your StorageGRID system is located under the configuration tab, Monitoring column. Enable SNMP and enter the desired information. If you wish to configure traps, select the "Traps Destinations" and Create a destination for the Datadog agent host containing the trap configuration.



Use rclone to migrate, PUT, and DELETE objects on StorageGRID

By Siegfried Hepp and Aron Klein

rclone is a free command line tool and client for S3 operations. You can use rclone to to migrate, copy, and delete object data on StorageGRID. rclone includes the capability to delete buckets even when not empty with a "purge" function as seen in an example below.

Install and configure rclone

To install rclone on a workstation or server, download it from rclone.org.

Initial configuration steps

1. Create the rclone configuration file by either running the config script or manually creating the file.
2. For this example I will use sgdemo for the name of the remote StorageGRID S3 endpoint in the rclone configuration.
   a. Create the config file ~/.config/rclone/rclone.conf

```
[sgdemo]
type = s3
provider = Other
access_key_id = ABCDEFGH123456789JKL
secret_access_key = 123456789ABCDEFGHIJKLMN0123456789PQRST+V
endpoint = sgdemo.netapp.com
```

   b. Run rclone config

rclone config

```
2023/04/13 14:22:45 NOTICE: Config file
"/root/.config/rclone/rclone.conf" not found - using defaults
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q> n
name> sgdemo
```

```
Option Storage.
Type of storage to configure.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value.
 1 / 1Fichier
   \ "fichier"
 2 / Alias for an existing remote
   \ "alias"
 3 / Amazon Drive
   \ "amazon cloud drive"
 4 / Amazon S3 Compliant Storage Providers including AWS,
Alibaba, Ceph, Digital Ocean, Dreamhost, IBM COS, Minio,
SeaweedFS, and Tencent COS
   \ "s3"
 5 / Backblaze B2
   \ "b2"
 6 / Better checksums for other remotes
   \ "hasher"
 7 / Box
   \ "box"
 8 / Cache a remote
   \ "cache"
 9 / Citrix Sharefile
   \ "sharefile"
10 / Compress a remote
   \ "compress"
11 / Dropbox
   \ "dropbox"
12 / Encrypt/Decrypt a remote
   \ "crypt"
13 / Enterprise File Fabric
   \ "filefabric"
14 / FTP Connection
```

```
   \ "ftp"
15 / Google Cloud Storage (this is not Google Drive)
   \ "google cloud storage"
16 / Google Drive
   \ "drive"
17 / Google Photos
   \ "google photos"
18 / Hadoop distributed file system
   \ "hdfs"
19 / Hubic
   \ "hubic"
20 / In memory object storage system.
   \ "memory"
21 / Jottacloud
   \ "jottacloud"
22 / Koofr
   \ "koofr"
23 / Local Disk
   \ "local"
24 / Mail.ru Cloud
   \ "mailru"
25 / Mega
   \ "mega"
26 / Microsoft Azure Blob Storage
   \ "azureblob"
27 / Microsoft OneDrive
   \ "onedrive"
28 / OpenDrive
   \ "opendrive"
29 / OpenStack Swift (Rackspace Cloud Files, Memset Memstore,
OVH)
   \ "swift"
30 / Pcloud
   \ "pcloud"
31 / Put.io
   \ "putio"
32 / QingCloud Object Storage
   \ "qingstor"
33 / SSH/SFTP Connection
   \ "sftp"
34 / Sia Decentralized Cloud
   \ "sia"
35 / Sugarsync
   \ "sugarsync"
36 / Tardigrade Decentralized Cloud Storage
   \ "tardigrade"
```

```
37 / Transparently chunk/split large files
   \ "chunker"
38 / Union merges the contents of several upstream fs
   \ "union"
39 / Uptobox
   \ "uptobox"
40 / Webdav
   \ "webdav"
41 / Yandex Disk
   \ "yandex"
42 / Zoho
   \ "zoho"
43 / http Connection
   \ "http"
44 / premiumize.me
   \ "premiumizeme"
45 / seafile
   \ "seafile"
```

```
Storage> 4
```

```
Option provider.
Choose your S3 provider.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value.
 1 / Amazon Web Services (AWS) S3
   \ "AWS"
 2 / Alibaba Cloud Object Storage System (OSS) formerly Aliyun
   \ "Alibaba"
 3 / Ceph Object Storage
   \ "Ceph"
 4 / Digital Ocean Spaces
   \ "DigitalOcean"
 5 / Dreamhost DreamObjects
   \ "Dreamhost"
 6 / IBM COS S3
   \ "IBMCOS"
 7 / Minio Object Storage
   \ "Minio"
 8 / Netease Object Storage (NOS)
   \ "Netease"
 9 / Scaleway Object Storage
   \ "Scaleway"
10 / SeaweedFS S3
   \ "SeaweedFS"
11 / StackPath Object Storage
   \ "StackPath"
12 / Tencent Cloud Object Storage (COS)
   \ "TencentCOS"
13 / Wasabi Object Storage
   \ "Wasabi"
14 / Any other S3 compatible provider
   \ "Other"
provider> 14
```

```
Option env_auth.
Get AWS credentials from runtime (environment variables or
EC2/ECS meta data if no env vars).
Only applies if access_key_id and secret_access_key is blank.
Enter a boolean value (true or false). Press Enter for the
default ("false").
Choose a number from below, or type in your own value.
 1 / Enter AWS credentials in the next step.
   \ "false"
 2 / Get AWS credentials from the environment (env vars or IAM).
   \ "true"
env_auth> 1
```

```
Option access_key_id.
AWS Access Key ID.
Leave blank for anonymous access or runtime credentials.
Enter a string value. Press Enter for the default ("").
access_key_id> ABCDEFGH123456789JKL
```

```
Option secret_access_key.
AWS Secret Access Key (password).
Leave blank for anonymous access or runtime credentials.
Enter a string value. Press Enter for the default ("").
secret_access_key> 123456789ABCDEFGHIJKLMN0123456789PQRST+V
```

```
Option region.
Region to connect to.
Leave blank if you are using an S3 clone and you don't have a
region.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value.
   / Use this if unsure.
 1 | Will use v4 signatures and an empty region.
   \ ""
   / Use this only if v4 signatures don't work.
 2 | E.g. pre Jewel/v10 CEPH.
   \ "other-v2-signature"
region> 1
```

```
Option endpoint.
Endpoint for S3 API.
Required when using an S3 clone.
Enter a string value. Press Enter for the default ("").
endpoint> sgdemo.netapp.com
```

```
Option location_constraint.
Location constraint - must be set to match the Region.
Leave blank if not sure. Used when creating buckets only.
Enter a string value. Press Enter for the default ("").
location_constraint>
```

```
Option acl.
Canned ACL used when creating buckets and storing or copying
objects.
This ACL is used for creating objects and if bucket_acl isn't
set, for creating buckets too.
For more info visit
https://docs.aws.amazon.com/AmazonS3/latest/dev/acl-
overview.html#canned-acl
Note that this ACL is applied when server-side copying objects as
S3
doesn't copy the ACL from the source but rather writes a fresh
one.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value.
   / Owner gets FULL_CONTROL.
 1 | No one else has access rights (default).
   \ "private"
   / Owner gets FULL_CONTROL.
 2 | The AllUsers group gets READ access.
   \ "public-read"
   / Owner gets FULL_CONTROL.
 3 | The AllUsers group gets READ and WRITE access.
   | Granting this on a bucket is generally not recommended.
   \ "public-read-write"
   / Owner gets FULL_CONTROL.
 4 | The AuthenticatedUsers group gets READ access.
   \ "authenticated-read"
   / Object owner gets FULL_CONTROL.
 5 | Bucket owner gets READ access.
   | If you specify this canned ACL when creating a bucket,
Amazon S3 ignores it.
   \ "bucket-owner-read"
   / Both the object owner and the bucket owner get FULL_CONTROL
over the object.
 6 | If you specify this canned ACL when creating a bucket,
Amazon S3 ignores it.
   \ "bucket-owner-full-control"
acl>
```

```
Edit advanced config?
y) Yes
n) No (default)
y/n> n
```

```
--------------------
[sgdemo]
type = s3
provider = Other
access_key_id = ABCDEFGH123456789JKL
secret_access_key = 123456789ABCDEFGHIJKLMN0123456789PQRST+V
endpoint = sgdemo.netapp.com:443
--------------------
y) Yes this is OK (default)
e) Edit this remote
d) Delete this remote
y/e/d>
```

```
Current remotes:
```

```
Name                 Type
====                 ====
sgdemo               s3
```

```
e) Edit existing remote
n) New remote
d) Delete remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
e/n/d/r/c/s/q> q
```

Basic command examples

- **Create a bucket:**

  `rclone mkdir remote:bucket`

  ```
  # rclone mkdir sgdemo:test01
  ```

  Use --no-check-certificate if you need to ignore SSL certificates.

- **List all buckets:**

```
rclone lsd remote:
```

```
# rclone lsd sgdemo:
```

- **List objects in a specific bucket:**

```
rclone ls remote:bucket
```

```
# rclone ls sgdemo:test01
```

```
    65536 TestObject.0
    65536 TestObject.1
    65536 TestObject.10
    65536 TestObject.12
    65536 TestObject.13
    65536 TestObject.14
    65536 TestObject.15
    65536 TestObject.16
    65536 TestObject.17
    65536 TestObject.18
    65536 TestObject.2
    65536 TestObject.3
    65536 TestObject.5
    65536 TestObject.6
    65536 TestObject.7
    65536 TestObject.8
    65536 TestObject.9
  33554432 bigobj
      102 key.json
       47 locked01.txt
4294967296 sequential-read.0.0
       15 test.txt
      116 version.txt
```

- **Delete a bucket:**

```
rclone rmdir remote:bucket
```

```
# rclone rmdir sgdemo:test02
```

- **Put an object:**

```
rclone copy filename remote:bucket
```

```
# rclone copy ~/test/testfile.txt sgdemo:test01
```

- **Get an object:**

```
rclone copy remote:bucket/objectname filename
```

```
# rclone copy sgdemo:test01/testfile.txt ~/test/testfileS3.txt
```

- **Delete an object:**

```
rclone delete remote:bucket/objectname
```

```
# rclone delete sgdemo:test01/testfile.txt
```

- **Migrate objects in a bucket**

```
rclone sync source:bucket destination:bucket --progress

rclone sync source_directory destination:bucket --progress
```

```
# rclone sync sgdemo:test01 sgdemo:clone01 --progress

Transferred:        4.032 GiB / 4.032 GiB, 100%, 95.484 KiB/s, ETA
0s
Transferred:           22 / 22, 100%
Elapsed time:       1m4.2s
```

Use --progress or -P to display the progress of the task. Otherwise there is no output.

- **Delete a bucket and all object contents**

```
rclone purge remote:bucket --progress
```

rclone purge sgdemo:test01 --progress

```
Transferred:              0 B / 0 B, -, 0 B/s, ETA -
Checks:                46 / 46, 100%
Deleted:               23 (files), 1 (dirs)
Elapsed time:         10.2s
```

rclone ls sgdemo:test01

```
2023/04/14 09:40:51 Failed to ls: directory not found
```

StorageGRID best practices for deployment with Veeam Backup and Replication

By Oliver Haensel and Aron Klein

This guide focuses on the configuration of NetApp StorageGRID and partly Veeam Backup and Replication. This paper is written for storage and network administrators who are familiar with Linux systems and tasked with maintaining or implementing a NetApp StorageGRID system in combination with Veeam Backup and Replication.

Overview

Storage Administrators are looking to manage the growth of their data with solutions that will meet the availability, rapid recovery goals, scale to meet their needs and automate their policy for long-term retention of data. These solutions should also provide protection from loss or malicious attacks. Together, Veeam and NetApp have partnered to create a data protection solution combining Veeam Backup & Recovery with NetApp StorageGRID for on-premises object storage.

Veeam and NetApp StorageGRID provide an easy-to-use solution that work together to help meet the demands of rapid data growth and increasing regulations around the world. Cloud-based object storage is known for its resilience, ability to scale, operational and cost efficiencies that make it a natural choice as a target for your backups. This document will provide guidance and recommendations for the configuration of your Veeam Backup solution and StorageGRID system.

The object workload from Veeam creates a large number of concurrent PUT, DELETE, and LIST operations of small objects. Enabling immutability will add to the number of requests to the object store for setting retention and listing versions. The process of a backup job includes writing objects for the daily change then after the new writes are complete the job will delete any objects based on the retention policy of the backup. The scheduling of backup jobs will almost always overlap. This overlap will result in a large portion of the backup window consisting of 50/50 PUT/DELETE workload on the object store. Making adjustments in Veeam to the number of concurrent operations with the task slot setting, increasing the object size by increasing the backup job block size, reducing the number of objects in the multi-object delete requests, and choosing the maximum time window for the jobs to complete will optimize the solution for performance and cost.

Make sure to read the product documentation for Veeam Backup and Replication and StorageGRID before you begin. Veeam provides calculators for understanding the sizing of the Veeam infrastructure and capacity

requirements that should be used prior to sizing your StorageGRID solution. Please always check the Veeam-NetApp validated configurations at the Veeam Ready Program website for Veeam Ready Object, Object Immutability, and Repository.

Veeam configuration

Recommended version

It is always recommended to stay current and apply the latest hotfixes for your Veeam Backup & Replication 12 or 12.1 system. Currently we recommend at a minimum installing Veeam 12 patch P20230718.

S3 Repository configuration

A scale-out backup repository (SOBR) is the capacity tier of S3 object storage. The capacity tier is an extension of the primary repository providing longer data retention periods and a lower cost storage solution. Veeam offers the ability to provide immutability through the S3 Object Lock API. Veeam 12 can use multiple buckets in a scale out repository. StorageGRID does not have a limit for the number of objects or capacity in a single bucket. Using multiple buckets may improve performance when backing up very large datasets where the backup data could get to petabyte scale in objects.

Limiting concurrent tasks may be required depending on the sizing of your specific solution and requirements. The default settings specify one repository task slot for each CPU core and for each task slot a concurrent task slot limit of 64. For example if your server has 2 CPU cores a total of 128 concurrent threads will be used for the object store. This is inclusive of PUT, GET, and batch Delete. It is recommended to select a conservative limit to the task slots to start with and tune this value once Veeam backups have reached a steady state of new backups and expiring backup data. Please work with your NetApp account team to size the StorageGRID system appropriately to meet the desired time windows and performance. Adjusting the number of task slots and the limit of tasks per slot may be required to provide the optimal solution.

Backup job configuration

Veeam backup jobs can be configured with different block size options that should be considered carefully. The default block size is 1MB and with the storage efficiencies Veeam provides with compression and deduplication creates object sizes of approximately 500kB for the initial Full backup and 100-200kB objects for the incremental jobs. We can greatly increase performance and scale down the requirements for the object store by choosing a larger backup block size. Though the larger block size makes great improvements in the object store performance it comes at the cost of potentially increased primary storage capacity requirement due to reduced storage efficiency performance. It is recommended for the backup jobs to be configured with a 4MB block size which creates approximately 2MB objects for the full backups and 700kB-1MB object sizes for incrementals. Customers may consider even configuring backup jobs using 8 MB block size, which can be enabled with assistance from Veeam support.

The implementation of immutable backups makes use of S3 Object Lock on the object store. The immutability option generates an increased number of requests to the object store for listing and retention updates on the objects.

As backup retentions expire the backup jobs will process the deletion of objects. Veeam sends the delete requests to the object store in multi-object delete requests of 1000 objects per request. For small solutions this may need to be adjusted to reduce the number of objects per request. Lowering this value will have the added benefit of more evenly distributing the delete requests across the nodes in the StorageGRID system. It is recommended to use the values in the table below as a starting point in configuring the multi object delete limit. Multiply the value in the table by the number of nodes for the chosen appliance type to get the value for the setting in Veeam. If this value is equal to or greater than 1000 there is no need to adjust the default value. If this value needs to be adjusted, please work with Veeam support to make the change.

Appliance Model	S3MultiObjectDeleteLimit per node
SG5712	34
SG5760	75
SG6060	200

Please work with your NetApp Account team for the recommended configuration based on your specific needs. The Veeam configuration settings recommendations will include:

- Backup job block size = 4MB
- SOBR task slot limit= 2-16
- Multi Object Delete Limit = 34-1000

StorageGRID configuration

Recommended version

NetApp StorageGRID 11.9 or 12.0 with the latest hotfix are the recommended versions for Veeam deployments. It is always recommended to stay current and apply the latest hotfixes for your StorageGRID system.

Load balancer and S3 endpoint configuration

Veeam requires the endpoint to be connected via HTTPS only. A non-encrypted connection is not supported by Veeam. The SSL certificate can be a self-signed certificate, private trusted certificate authority, or public trusted certificate authority. To ensure continuous access to the S3 repository it is recommended to use at least two load balancers in an HA configuration. The load balancers can be a StorageGRID provided integrated load balancer service located on every admin node and gateway node or third-party solution such as F5, Kemp, HAproxy, Loadbalancer.org, etc. Using a StorageGRID load balancer will provide the ability to set traffic classifiers (QoS rules) that can prioritize the Veeam workload, or limit Veeam to not impact higher priority workloads on the StorageGRID system.

S3 Bucket

StorageGRID is a secure multi-tenant storage system. It is recommended to create a dedicated tenant for the Veeam workload. A storage quota can be optionally assigned. As a best practice enable “use own identity source”. Secure the tenant root management user with an appropriate password. Veeam Backup 12 requires strong consistency for S3 buckets. StorageGRID offers multiple consistency options configured at the bucket level. For multi-site deployments with Veeam accessing the data from multiple locations, select “strong-global”. If Veeam backups and restores happen at a single site only, consistency level should be set to “strong-site”. For more information on bucket consistency levels please review the documentation. To use StorageGRID for Veeam immutability backups, S3 Object Lock must be enabled globally and configured on the bucket during the bucket creation.

Lifecycle management

StorageGRID supports replication and erasure coding for object level protection across StorageGRID nodes and sites. Erasure Coding requires at least a 200kB object size. The default block size for Veeam of 1MB produces object sizes that can often be below this 200kB recommended minimum size after Veeam’s storage efficiencies. For the performance of the solution, it is not recommended to use an erasure coding profile spanning multiple sites unless the connectivity between the sites is sufficient to not add latency or restrict the

bandwidth of the StorageGRID system. In a multi-site StorageGRID system the ILM rule can be configured to store a single copy at each site. For ultimate durability a rule could be configured to store an erasure coded copy at each site. Using two copies local to the Veeam Backup servers is the most recommended implementation for this workload.

Delete performance

Veeam provides delete request rate tuning and scheduling of the backup delete process. To further tune delete performance you can disable synchronous deletes and let the ILM scanner manage the eventual deletion of objects.

Steps to Disabling Synchronous Deletes

1. Open the StorageGRID Grid Manager.
2. In top right corner, select the Question mark then API Documentation.
3. In top right corner, click on the Private API Documentation page link.
4. Expand ilm-advanced.
5. Select GET ilm-advanced.
6. Select Try it out, then Execute.
7. Check the response result.
    a. If the values are null, then it means the default ilm-advanced values are in-use.
    b. If the values are not null, then it means custom ILM advanced values are in-use. Copy all the output after "data" :, starting with the { up untill the second to last }.
        i. Save it in some text editor.

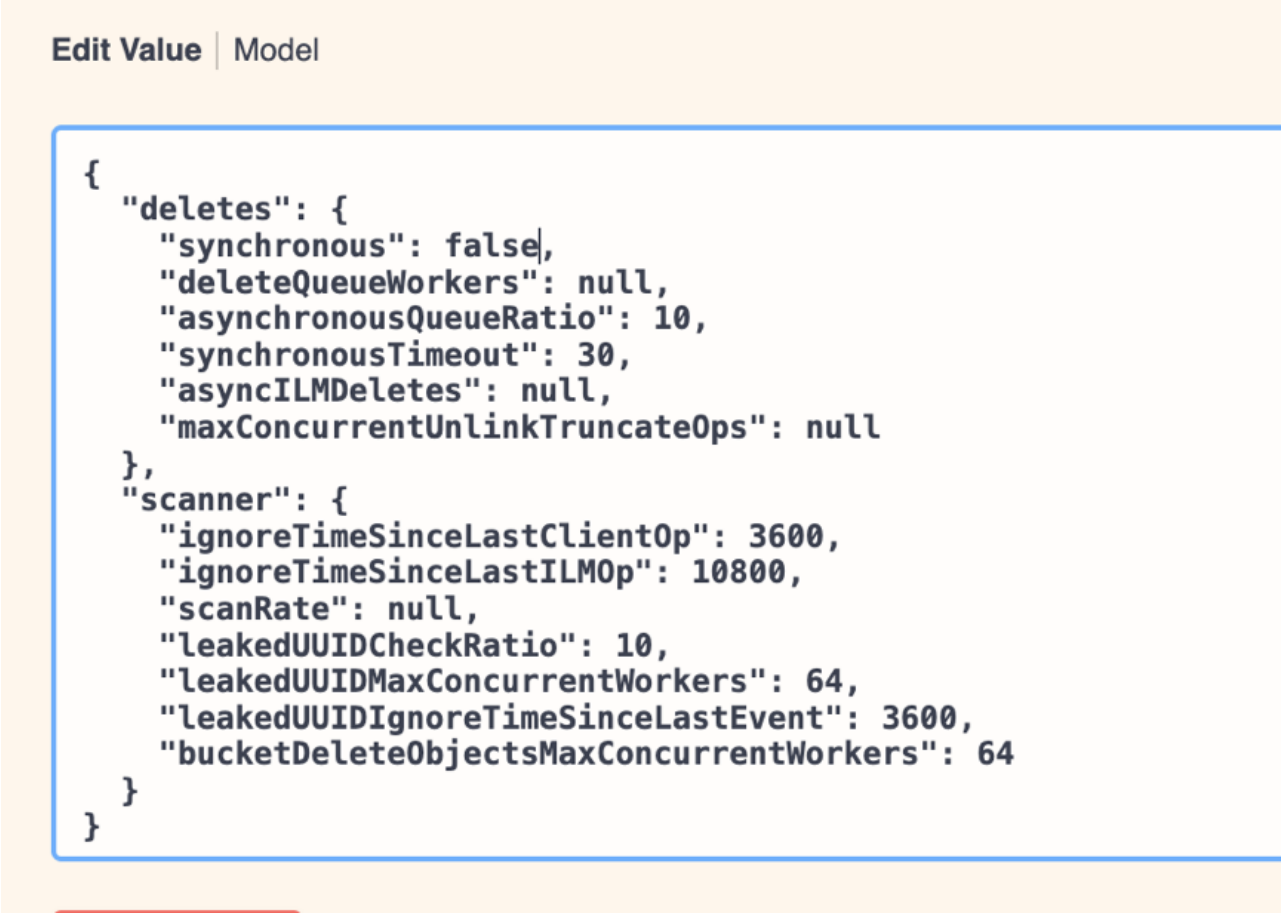
        Example response:

Response body


```
{
  "responseTime": "2025-09-19T15:01:28.142Z",
  "status": "success",
  "apiVersion": "4.2",
  "data": {
    "deletes": {
      "synchronous": null,
      "deleteQueueWorkers": null,
      "asynchronousQueueRatio": null,
      "synchronousTimeout": null,
      "asyncILMDeletes": null,
      "maxConcurrentUnlinkTruncateOps": null
    },
    "scanner": {
      "ignoreTimeSinceLastClientOp": null,
      "ignoreTimeSinceLastILMOp": null,
      "scanRate": null,
      "leakedUUIDCheckRatio": null,
      "leakedUUIDMaxConcurrentWorkers": null,
      "leakedUUIDIgnoreTimeSinceLastEvent": null,
      "bucketDeleteObjectsMaxConcurrentWorkers": null
    }
  }
}
```


8. Select PUT ilm-advanced.
9. Select Try it out to begin editing the API body.
    a. By default, the API body will contain default values and not any custom values that were previously configured. This is the reason it is VERY important to execute steps 5-7.
10. If non-default values are found in step 5-7, then replace the API body with the output saved in step 7. . Otherwise if the values were null in step 5-7, then leave API body as is.
11. Adjust the following parameters in the API body box:
    a. Set the synchronous value to false.

       Example API body text:

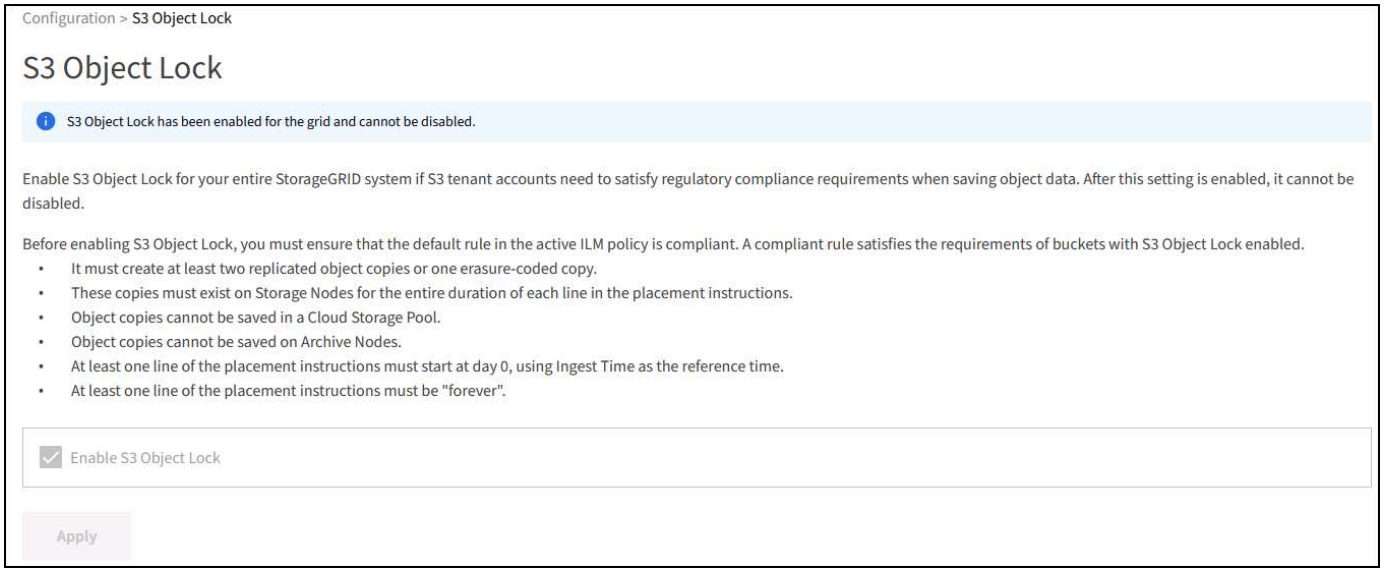


12. Once complete, select Execute

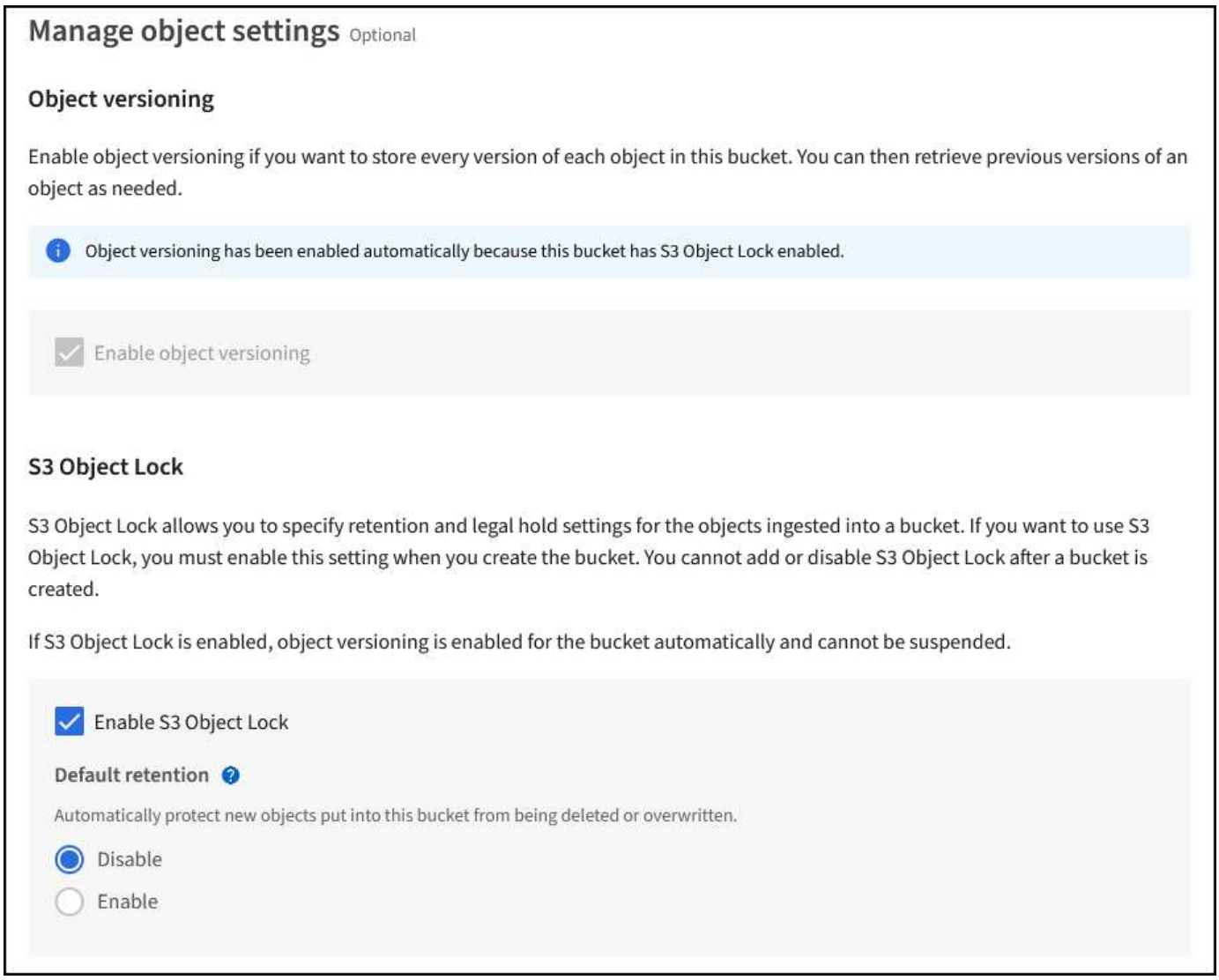
Implementation key points

StorageGRID

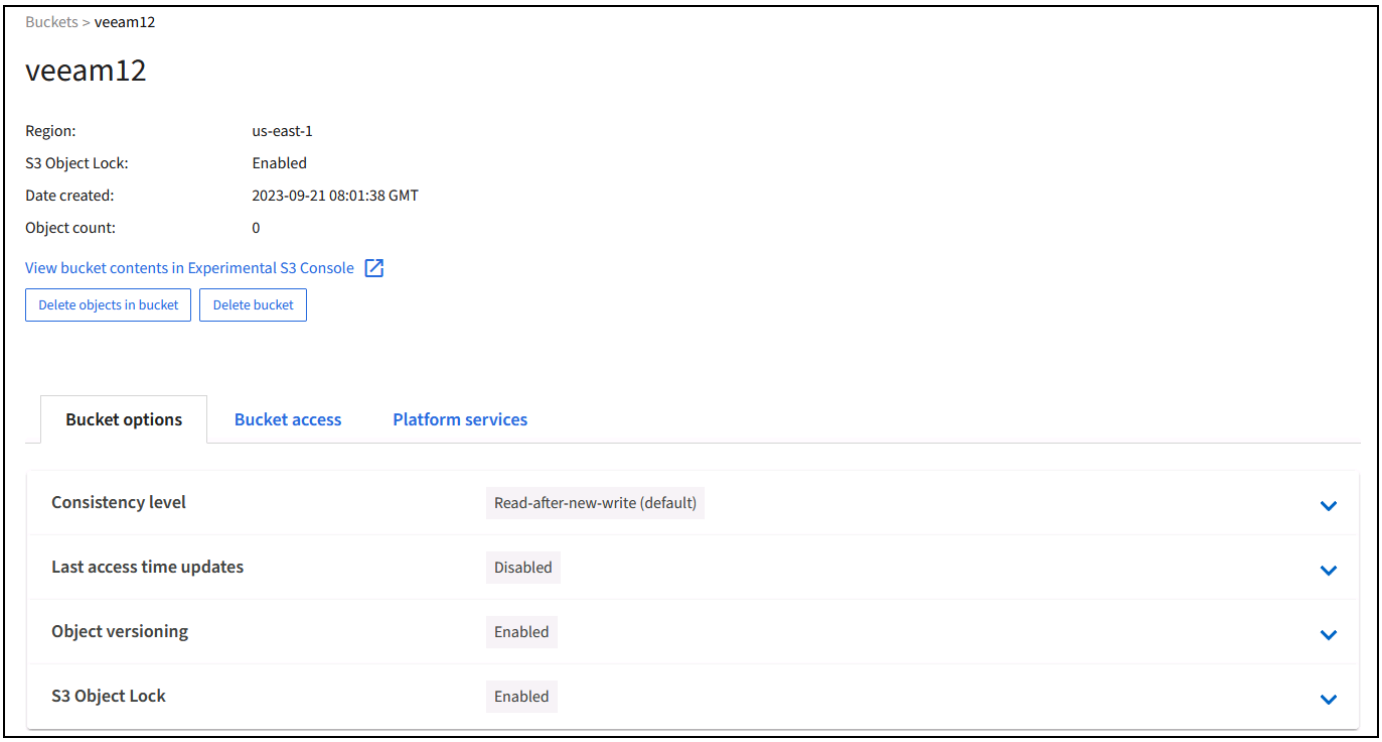
Ensure Object Lock is enabled on the StorageGRID system if immutability is required. Find the option in the management UI under Configuration/S3 Object Lock.



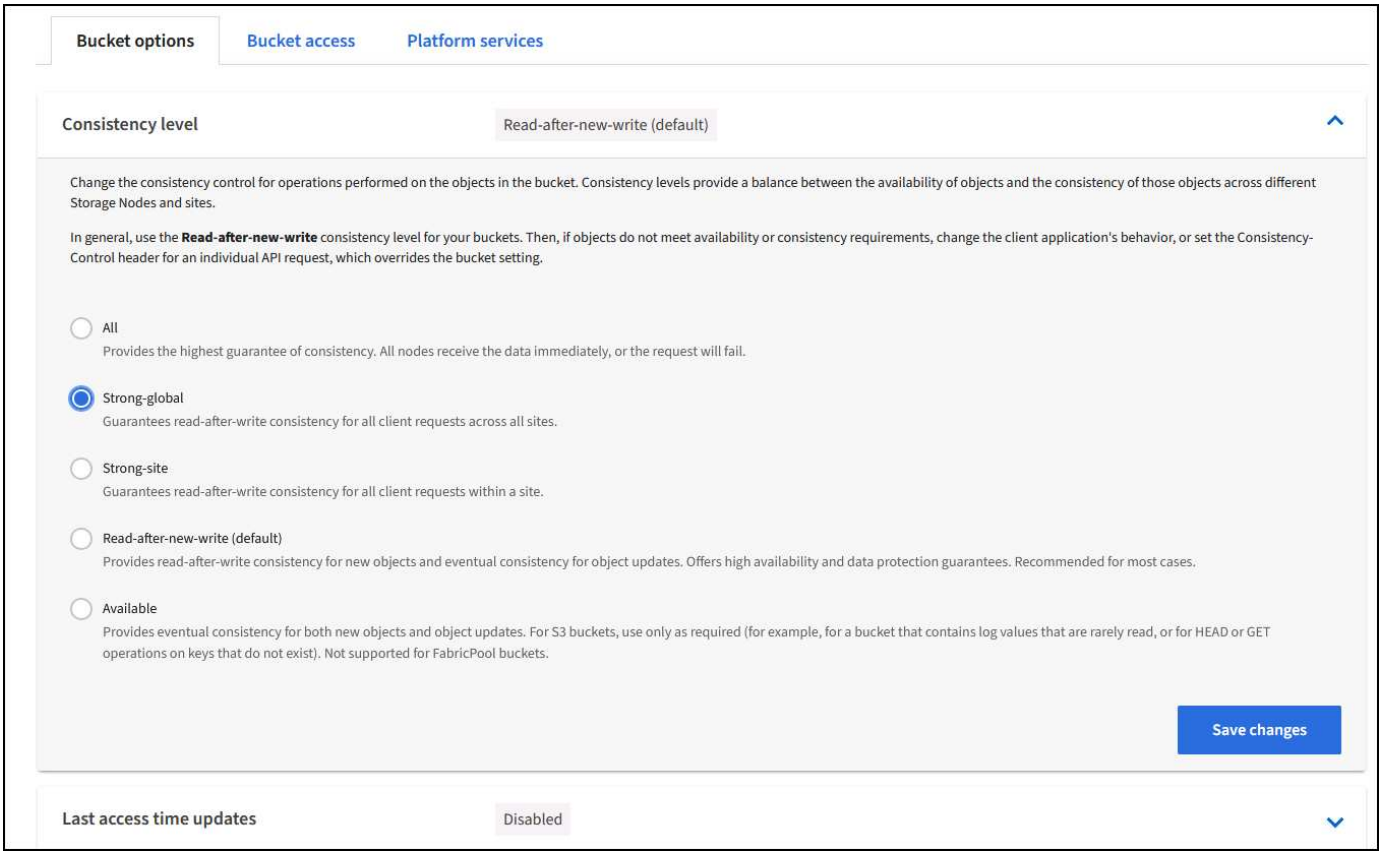
When creating the bucket, select “Enable S3 Object Lock” if this bucket is to be used for immutability backups. This will automatically enable bucket versioning. Leave default retention disabled as Veeam will set object retention explicitly. Versioning and S3 Object Lock should not be selected if Veeam isn’t creating immutable backups.



Once the bucket is created go to the details page of the bucket created. Select the consistency level.



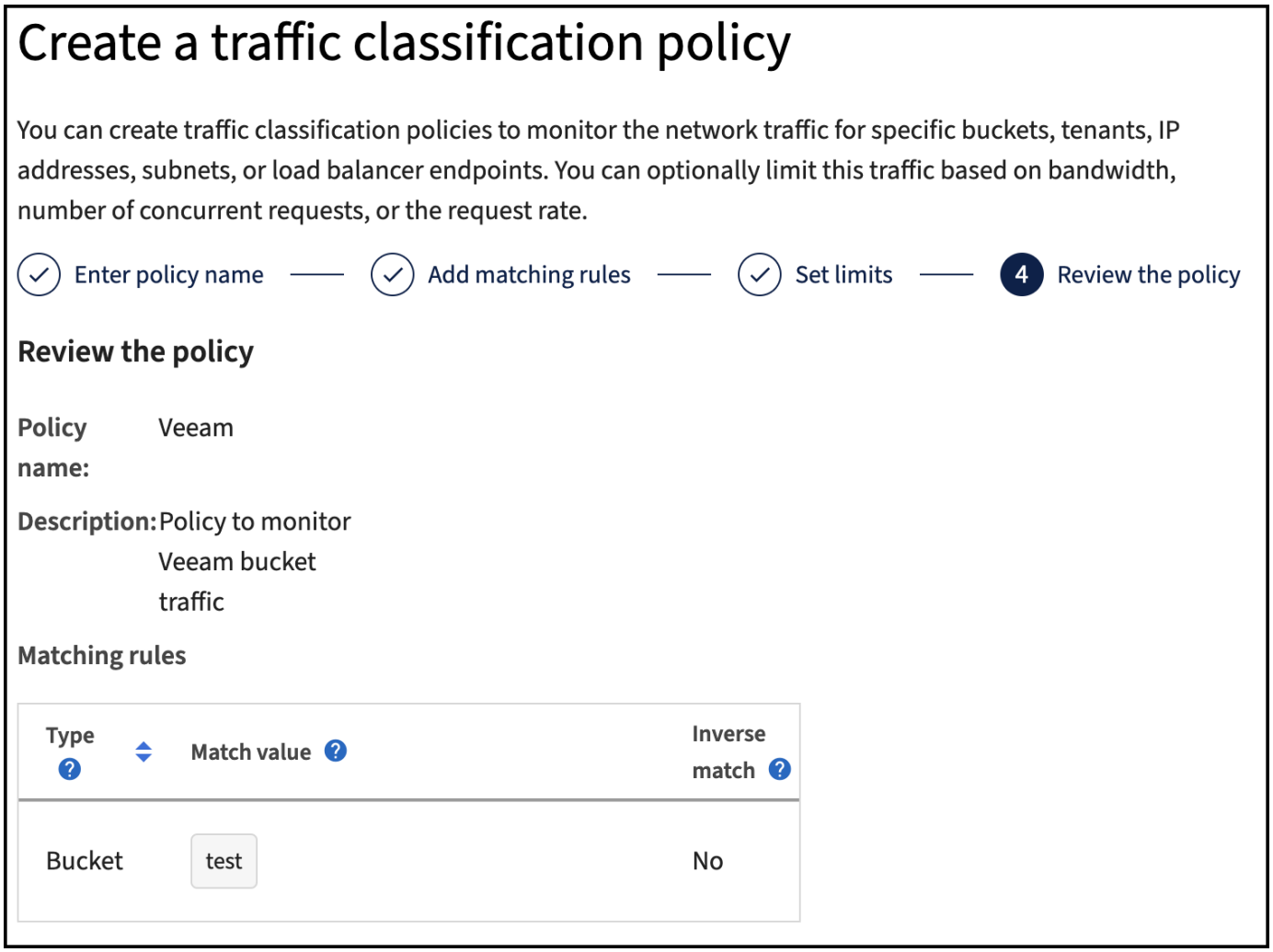
Veeam requires strong consistency for S3 buckets. So, for multi-site deployments with Veeam accessing the data from multiple locations, select “strong-global”. If Veeam backups and restores happen at a single site only, consistency level should be set to “strong-site”. Save the changes.



StorageGRID provides an integrated load balancer service on every admin node and dedicated gateway nodes. One of the many advantages of using this load balancer is the ability to configure Traffic Classification

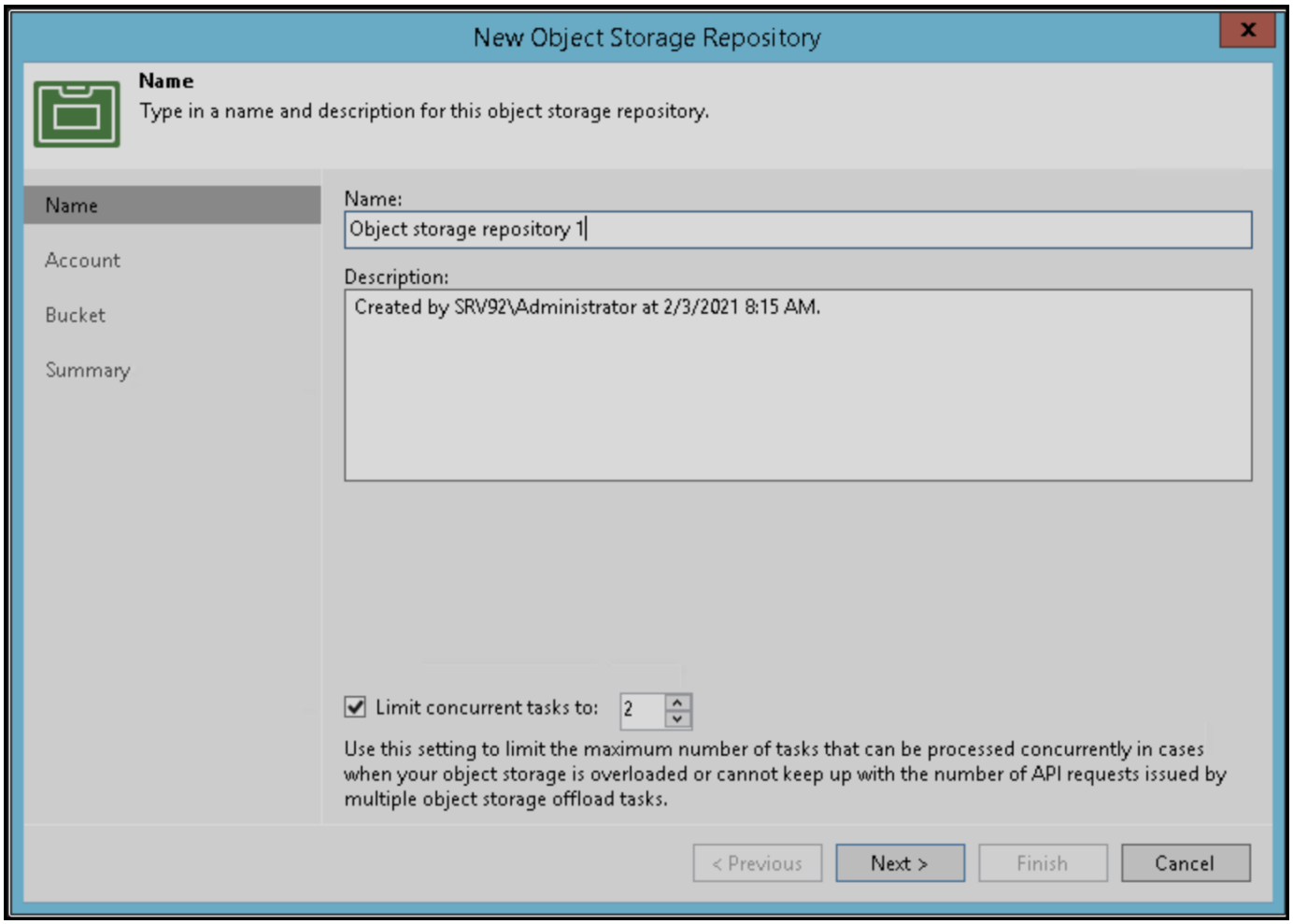
Policies (QoS). Though these are mainly used for limiting an applications impact on other client workloads or prioritizing a workload over others, they also provide a bonus of additional metrics collection to assist in monitoring.

In the configuration tab, select “Traffic Classification” and create a new policy. Name the rule and select either the bucket(s) or tenant as the type. Enter the name(s) of the bucket(s) or tenant. If QoS is required, set a limit, but for most implementations, we just want to add the monitoring benefits this provides so do not set a limit.

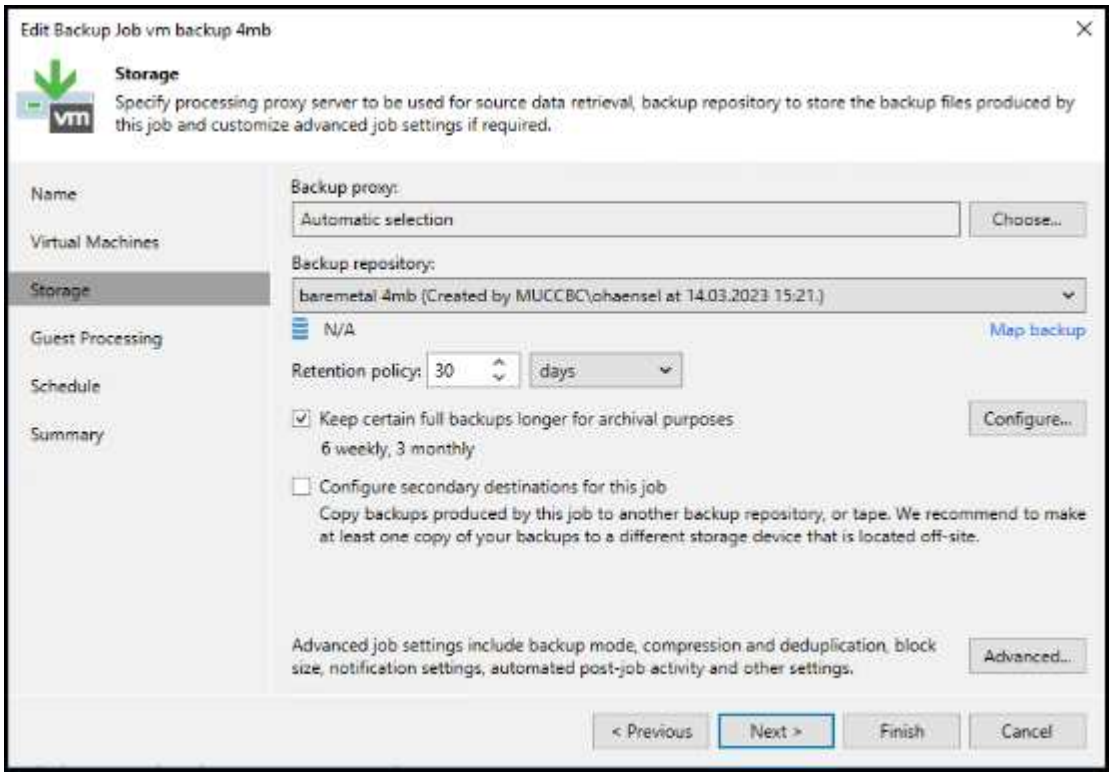


Veeam

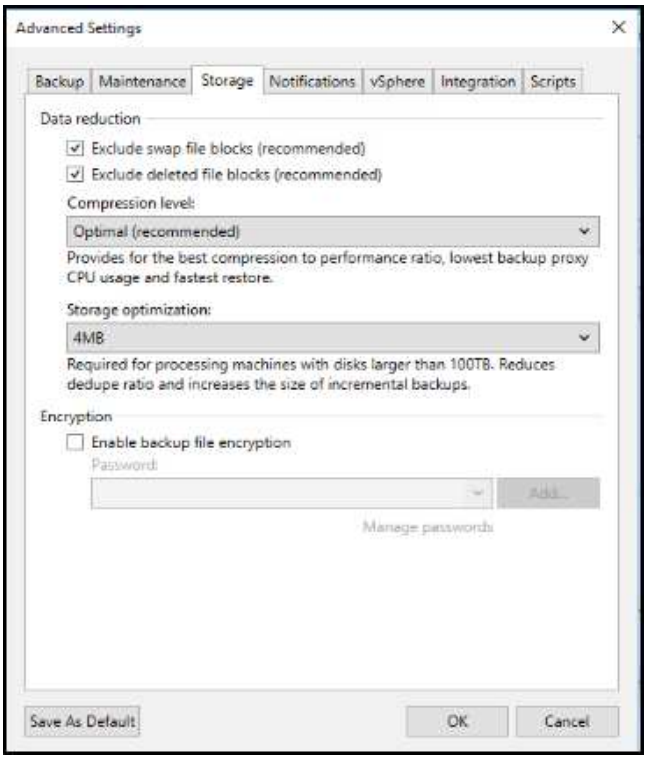
Depending on the model and quantity of StorageGRID appliances it may be necessary to select and configure a limit to the number of concurrent operations on the bucket.



Follow the Veeam documentation on backup job configuration in the Veeam console to start the wizard. After adding VMs select the SOBR repository.



Click Advanced settings and change storage optimization settings to 4 MB or larger. Compression and deduplication shall be enabled. Change guest settings according to your requirements and configure the backup job schedule.



Monitoring StorageGRID

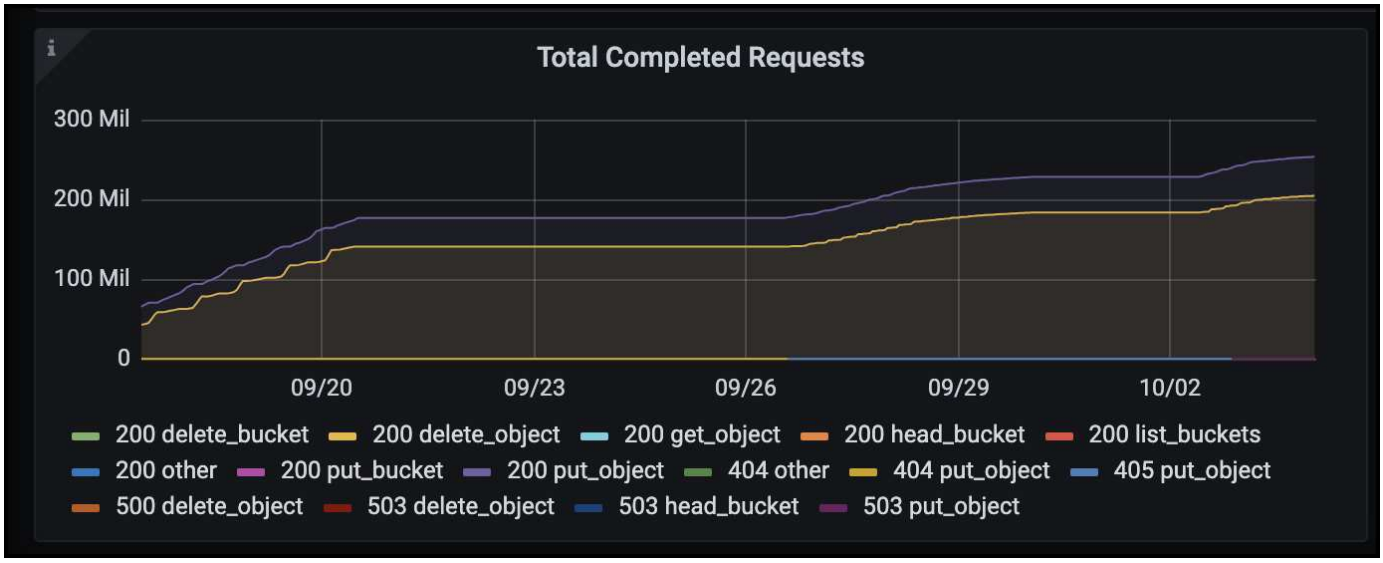
To get the full picture of how Veeam and StorageGRID are performing together you will need to wait until the retention time of the first backups have expired. Up until this point the Veeam workload consists primarily of PUT operations and no DELETEs have occurred. Once there is backup data expiring and cleanups are occurring you can now see the full consistent usage in the object store and adjust the settings in Veeam if needed.

StorageGRID provides convenient charts to monitor the operation of the system located in the Support tab Metrics page. The primary dashboards to look at will be the S3 Overview, ILM, and Traffic Classification Policy if a policy was created. In the S3 Overview dashboard you will find information on the S3 operation rates, latencies, and request responses.

Looking at the S3 rates and active requests you can see how much of the load each node is handling and the overall number of requests by type.

The Average Duration chart shows the average time each node is taking for each request type. This is the average latency of the request and may be a good indicator that additional tuning may be required, or there is room for the StorageGRID system to take on more load.

In the Total Completed Requests chart, you can see the requests by type and response codes. If you see responses other than 200 (Ok) for the responses this may indicate an issue like the StorageGRID system is getting heavily loaded sending 503 (Slow Down) responses and some additional tuning may be necessary, or the time has come to expand the system for the increased load.



In the ILM Dashboard you can monitor the Delete performance of your StorageGRID system. StorageGRID uses a combination of synchronous and asynchronous deletes on each node to try and optimize the overall performance for all requests.

With a Traffic Classification Policy, we can view metrics on the load balancer Request throughput, rates, duration, as well as the object sizes Veeam is sending and receiving.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp StorageGRID Product Documentation
- Veeam Backup and Replication

Configure Dremio data source with StorageGRID

By Angela Cheng

Dremio supports a varity of data sources, including cloud-based or on-premises object storage. You can configure Dremio to use StorageGRID as object storage data source.

Configure Dremio data source

Prerequisites

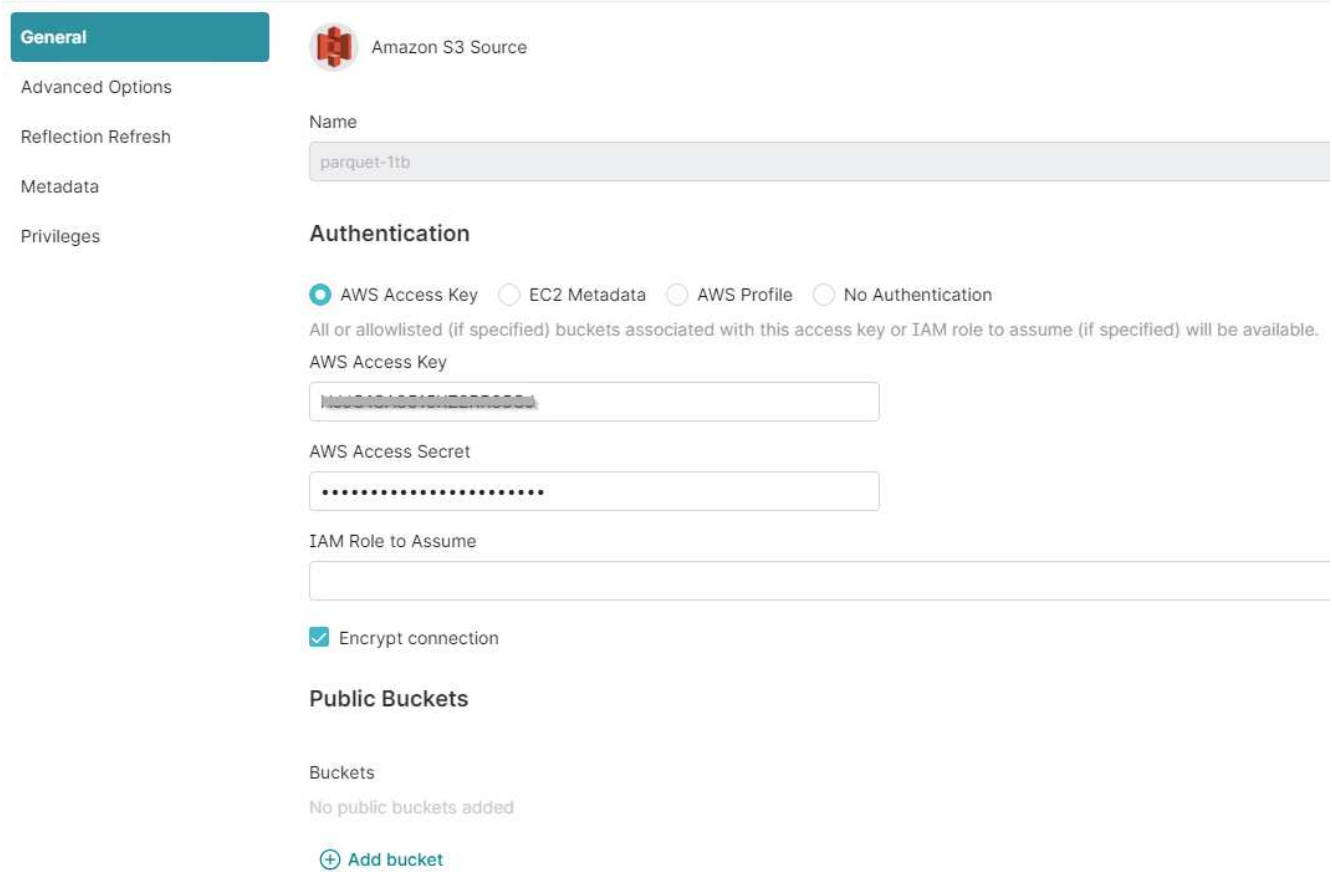
- A StorageGRID S3 endpoint URL, a tenant s3 access key ID, and secret access key.
- StorageGRID configuration recommendation: disable compression (disabled by default).
  Dremio uses byte range GET to fetch different byte ranges from within the same object concurrently during query. Typical size for byte-range requests is 1MB. Compressed object degrades byte-range GET performance.

Dremio guide

Connecting to Amazon S3 - Configuring S3-Compatible Storage.

Instruction

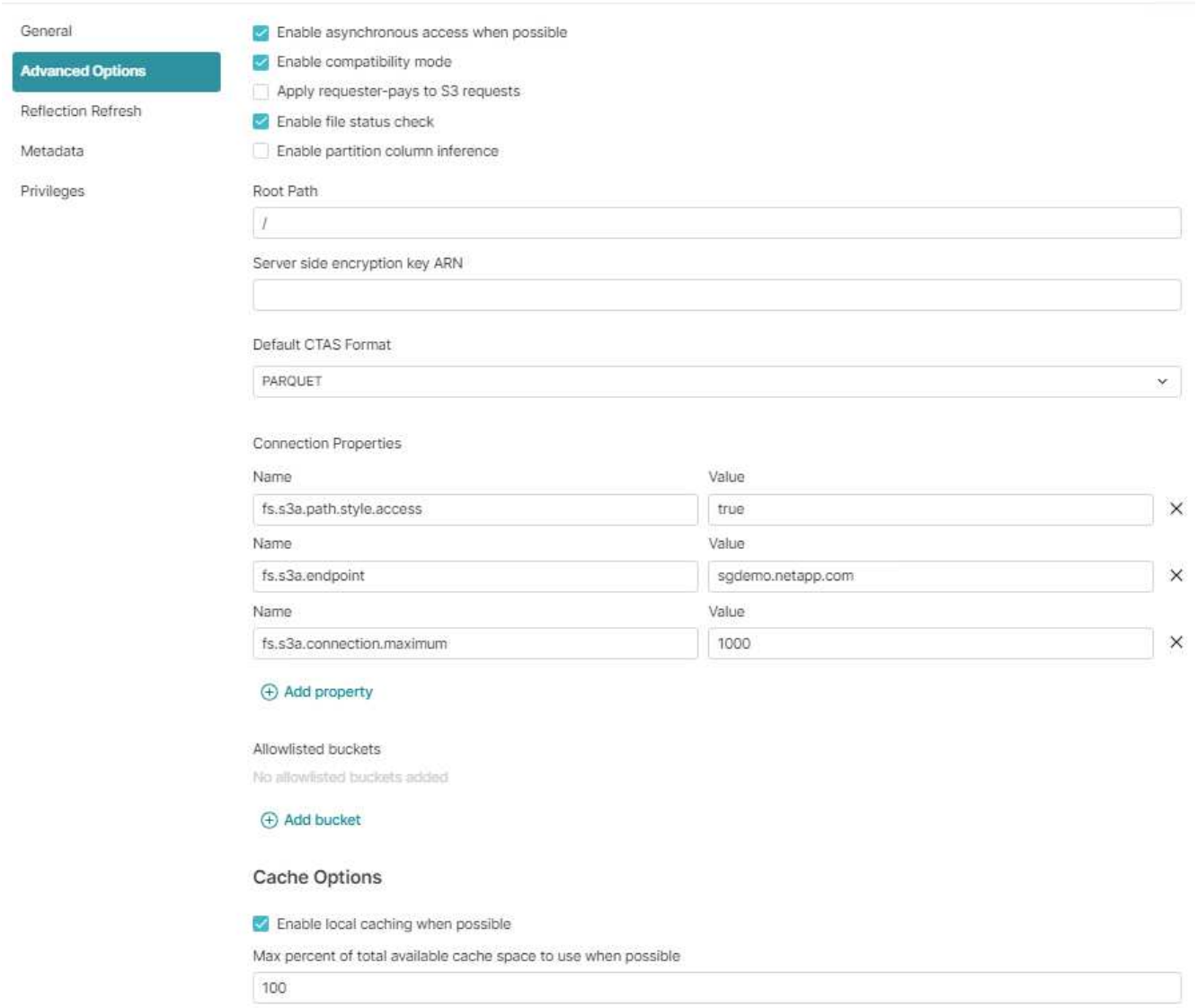
1. On Dremio Datasets page, click + sign to add a source, select 'Amazon S3'.
2. Enter a name for this new data source, StorageGRID S3 tenant access key ID and secret access key.
3. Check the box 'Encrypt connection' if using https for connection to StorageGRID S3 endpoint.
   If using self-signed CA cert for this s3 endpoint, follow Dremio guide instrution to add this CA cert into Dremio server's <JAVA_HOME>/jre/lib/security
   Sample screenshot



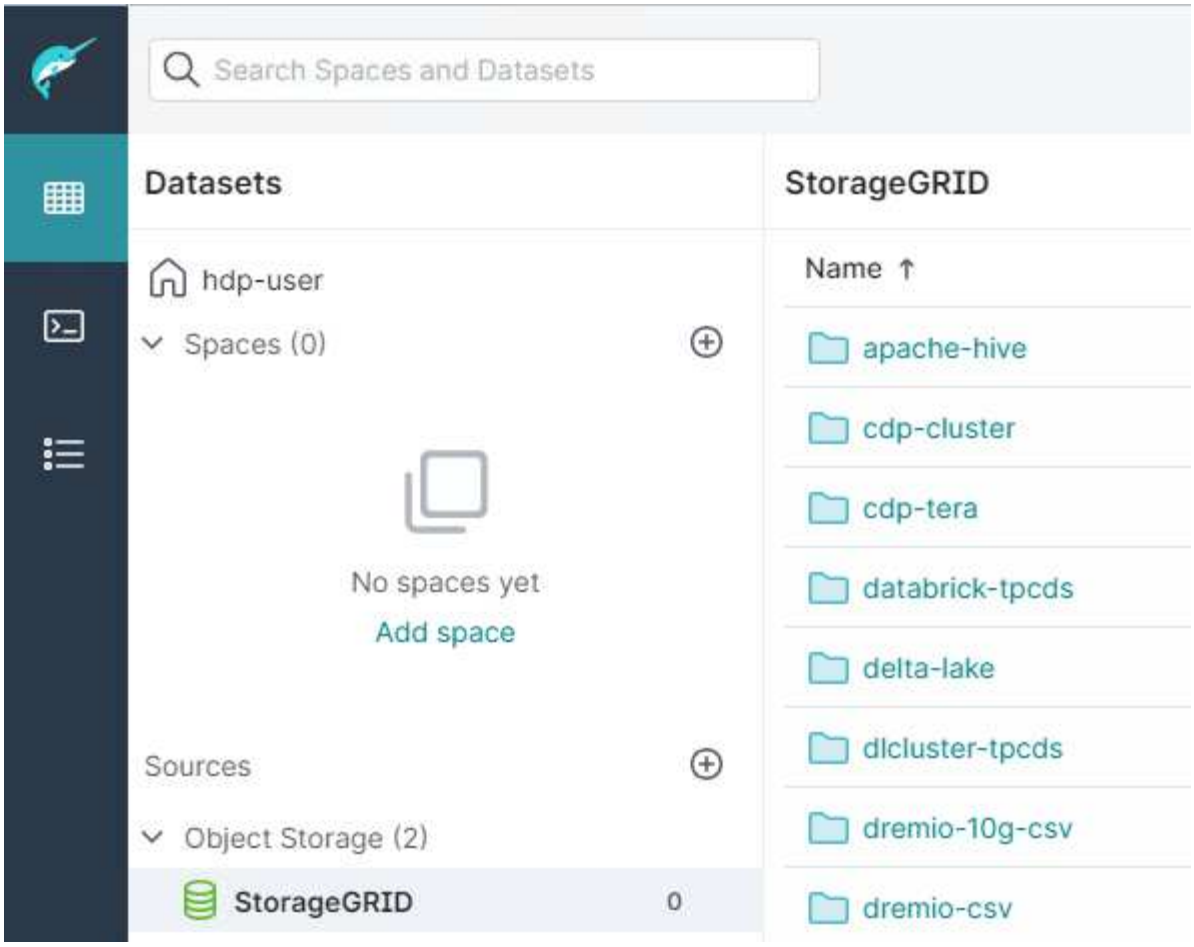
4. Click 'Advanced Options', check 'Enable compatibility mode'
5. Under Connection properties, click + Add Properties and add these s3a properties.
6. fs.s3a.connection.maximum default is 100. If your s3 datasets include large Parquet files with 100 or more columns, must enter a value greater than 100. Refer to Dremio guide for this setting.

Name	Value
fs.s3a.endpoint	*<StorageGRID S3 endpoint:port>*
fs.s3a.path.style.access	true
fs.s3a.connection.maximum	*<a value greater than 100>*

Sample screenshot



7. Configure other Dremio options as per your organization or application requirements.
8. Click the Save button to create this new data source.
9. Once StorageGRID data source is added successfully, a list of buckets will be displayed on the left panel.
Sample screenshot



NetApp StorageGRID with GitLab

By Angela Cheng

NetApp has tested StorageGRID with GitLab. See sample GitLab configuration below. Refer to GitLab object storage configuration guide for details.

Object Storage connection example

For Linux Package installations, this is an example of the `connection` setting in the consolidated form. Edit `/etc/gitlab/gitlab.rb` and add the following lines, substituting the values you want:

```
# Consolidated object storage configuration
gitlab_rails['object_store']['enabled'] = true
gitlab_rails['object_store']['proxy_download'] = true
gitlab_rails['object_store']['connection'] = {
  'provider' => 'AWS',
  'region' => 'us-east-1',
  'endpoint' => 'https://<storagegrid-s3-endpoint:port>',
  'path_stype' => 'true',
  'aws_access_key_id' => '<AWS_ACCESS_KEY_ID>',
  'aws_secret_access_key' => '<AWS_SECRET_ACCESS_KEY>'
}
# OPTIONAL: The following lines are only needed if server side encryption
is required
gitlab_rails['object_store']['storage_options'] = {
  'server_side_encryption' => 'AES256'
}
gitlab_rails['object_store']['objects']['artifacts']['bucket'] = 'gitlab-
artifacts'
gitlab_rails['object_store']['objects']['external_diffs']['bucket'] =
'gitlab-mr-diffs'
gitlab_rails['object_store']['objects']['lfs']['bucket'] = 'gitlab-lfs'
gitlab_rails['object_store']['objects']['uploads']['bucket'] = 'gitlab-
uploads'
gitlab_rails['object_store']['objects']['packages']['bucket'] = 'gitlab-
packages'
gitlab_rails['object_store']['objects']['dependency_proxy']['bucket'] =
'gitlab-dependency-proxy'
gitlab_rails['object_store']['objects']['terraform_state']['bucket'] =
'gitlab-terraform-state'
gitlab_rails['object_store']['objects']['pages']['bucket'] = 'gitlab-
pages'
```