Cargas de trabajo de Apache Kafka con almacenamiento NFS de NetApp

NetApp Solutions

NetApp
May 07, 2024

This PDF was generated from https://docs.netapp.com/es-es/netapp-solutions/data-analytics/kafka-nfs-introduction.html on May 07, 2024. Always check docs.netapp.com for the latest.

Tabla de contenidos

Cargas de trabajo de Apache Kafka con almacenamiento NFS de NetApp . . . 1
TR-4947: Carga de trabajo de Apache Kafka con almacenamiento NFS de NetApp: Validación y rendimiento funcionales . . . 1
Solución de NetApp para el problema del cambio de nombre más tonto de las cargas de trabajo de NFS a Kafka . . . 2
Validación funcional: Corrección de nombre de Silly . . . 3
¿Por qué NFS de NetApp para las cargas de trabajo de Kafka? . . . 8
Información general y validación del rendimiento en AWS . . . 20
Información general sobre el rendimiento y la validación en AWS FSx para NetApp ONTAP . . . 33
Descripción general y validación del rendimiento con AFF A900 en las instalaciones . . . 41
Conclusión . . . 48
Dónde encontrar información adicional . . . 48

Cargas de trabajo de Apache Kafka con almacenamiento NFS de NetApp

TR-4947: Carga de trabajo de Apache Kafka con almacenamiento NFS de NetApp: Validación y rendimiento funcionales

Shantanu Chakole, Karthikeyan Nagalingam y Joe Scott, NetApp

Kafka es un sistema de mensajería distribuida de suscripción a publicación con una cola sólida que puede aceptar grandes cantidades de datos de mensajes. Con Kafka, las aplicaciones pueden escribir y leer datos en los temas de un modo muy rápido. Debido a su tolerancia a fallos y su escalabilidad, Kafka se utiliza a menudo en el espacio de Big Data como una forma fiable de procesar y mover muchos flujos de datos muy rápidamente. Entre los casos de uso se incluyen el procesamiento de flujos, el seguimiento de la actividad de sitios web, la recopilación y supervisión de métricas, la agregación de registros, el análisis en tiempo real, etc.

Aunque las operaciones normales de Kafka en NFS funcionan bien, el "tonto renombrar" Issue bloquea la aplicación durante el cambio de tamaño o la partición de un clúster Kafka que se ejecuta en NFS. Esto es un problema significativo debido a que debe cambiarse el tamaño de un clúster Kafka o reparticionarse para fines de mantenimiento o equilibrio de carga. Puede encontrar más información "aquí".

Este documento describe los siguientes temas:

- El problema tonto-renombrar y la validación de solución
- Reducción del uso de CPU para reducir el tiempo de espera de I/O.
- Tiempo de recuperación más rápido de los agentes Kafka
- Rendimiento en el cloud y en las instalaciones

¿Por qué utilizar almacenamiento NFS para las cargas de trabajo de Kafka?

Las cargas de trabajo Kafka en aplicaciones de producción pueden transmitir enormes cantidades de datos entre aplicaciones. Estos datos se guardan y almacenan en los nodos de agente de Kafka en el clúster de Kafka. Kafka también se conoce por la disponibilidad y el paralelismo, que logra al dividir temas en particiones y luego replicarlas en el clúster. Esto eventualmente significa que la enorme cantidad de datos que fluye por un clúster de Kafka se multiplica por lo general en tamaño. NFS hace que el reequilibrio de datos a medida que el número de agentes cambia sea muy rápido y sencillo. En entornos de gran tamaño, hay que reequilibrar los datos en DAS cuando cambia el número de intermediarios y, en la mayoría de los entornos Kafka, el número de agentes cambia con frecuencia.

Entre otras ventajas, se incluyen las siguientes:

- **Madurez.** NFS es un protocolo maduro, que significa que la mayoría de los aspectos de implementación, seguridad y uso son bien entendidos.
- **Open.** NFS es un protocolo abierto, y su continuo desarrollo está documentado en las especificaciones de Internet como un protocolo de red libre y abierto.

- **Rentable.** NFS es una solución de bajo coste para compartir archivos de red que es fácil de configurar porque utiliza la infraestructura de red existente.
- **Gestión centralizada.** la administración centralizada de NFS reduce la necesidad de añadir software y espacio en disco en sistemas de usuario individuales.
- **Distributed.** NFS se puede utilizar como un sistema de archivos distribuido, reduciendo la necesidad de dispositivos de almacenamiento multimedia extraíbles.

¿Por qué elegir NetApp para las cargas de trabajo de Kafka?

La implantación de NFS de NetApp se considera un estándar oro para el protocolo y se utiliza en innumerables entornos NAS empresariales. Además de la credibilidad de NetApp, también ofrece las siguientes ventajas:

- Fiabilidad y eficiencia
- Escalabilidad y rendimiento
- Alta disponibilidad (partner de alta disponibilidad en un clúster ONTAP de NetApp)
- Protección de datos
  - **Recuperación ante desastres (SnapMirror de NetApp).** su sitio se cae o quiere empezar desde otro sitio y continuar desde el punto de partida.
  - Capacidad de gestión del sistema de almacenamiento (administración y gestión con OnCommand de NetApp).
  - **Equilibrio de carga.** el clúster le permite acceder a diferentes volúmenes desde LIF de datos alojadas en diferentes nodos.
  - **Operaciones no disruptivas.** los LIF o movimientos de volúmenes son transparentes para los clientes NFS.

Solución de NetApp para el problema del cambio de nombre más tonto de las cargas de trabajo de NFS a Kafka

Kafka se construye con la suposición de que el sistema de ficheros subyacente es compatible con POSIX: Por ejemplo, XFS o Ext4. El reequilibrio de recursos de Kafka elimina los archivos mientras la aplicación se sigue utilizando. Un sistema de archivos compatible con POSIX permite que se realice la desvinculación. Sin embargo, sólo quita el archivo una vez que todas las referencias al archivo hayan desaparecido. Si el sistema de archivos subyacente está conectado a la red, el cliente NFS intercepta las llamadas de desenlace y administra el flujo de trabajo. Dado que hay abiertos pendientes en el archivo que se está desvinculando, el cliente NFS envía una solicitud de cambio de nombre al servidor NFS y, en el último cierre del archivo sin vincular, emite una operación de eliminación en el archivo cuyo nombre ha cambiado. Este comportamiento se conoce comúnmente como nombre tonto de NFS y está orquestado por el cliente NFS.

Cualquier agente de Kafka que utilice almacenamiento de un servidor NFSv3 se ejecuta en problemas debido a este comportamiento. Sin embargo, el protocolo NFSv4.x tiene funciones para solucionar este problema permitiendo al servidor asumir la responsabilidad de los archivos abiertos sin vínculos. Los servidores NFS que admiten esta función opcional comunican la capacidad de propiedad al cliente NFS en el momento de abrir un archivo. A continuación, el cliente NFS detiene la gestión de desenlace cuando hay pendientes y

permite que el servidor gestione el flujo. Aunque la especificación NFSv4 proporciona directrices para la implementación, hasta ahora no hubo ninguna implementación conocida de servidores NFS que admita esta función opcional.

Para que el servidor NFS y el cliente NFS aborden el problema del cambio de nombre más tonto, son necesarios los siguientes cambios:

- **Cambios en el cliente NFS (Linux).** al abrir el archivo, el servidor NFS responde con un indicador, indicando la capacidad de manejar la desvinculación de los archivos abiertos. Los cambios en el cliente NFS permiten que el servidor NFS gestione el desvínculo en presencia del indicador. NetApp ha actualizado el cliente Linux NFS de código abierto con estos cambios. El cliente NFS actualizado ahora está disponible en general en RHEL8.7 y RHEL9.1.
- **Cambios en el servidor NFS.** el servidor NFS realiza un seguimiento de los abiertos. El servidor gestiona ahora la desvinculación de un archivo abierto existente para que coincida con la semántica POSIX. Cuando se cierra la última apertura, el servidor NFS inicia entonces la eliminación real del archivo y evita así el tonto proceso de cambio de nombre. El servidor NFS de ONTAP ha implementado esta funcionalidad en su última versión, ONTAP 9.12.1.

Con los cambios anteriores en el cliente y el servidor NFS, Kafka puede obtener de forma segura todas las ventajas del almacenamiento NFS conectado a la red.

Validación funcional: Corrección de nombre de Silly

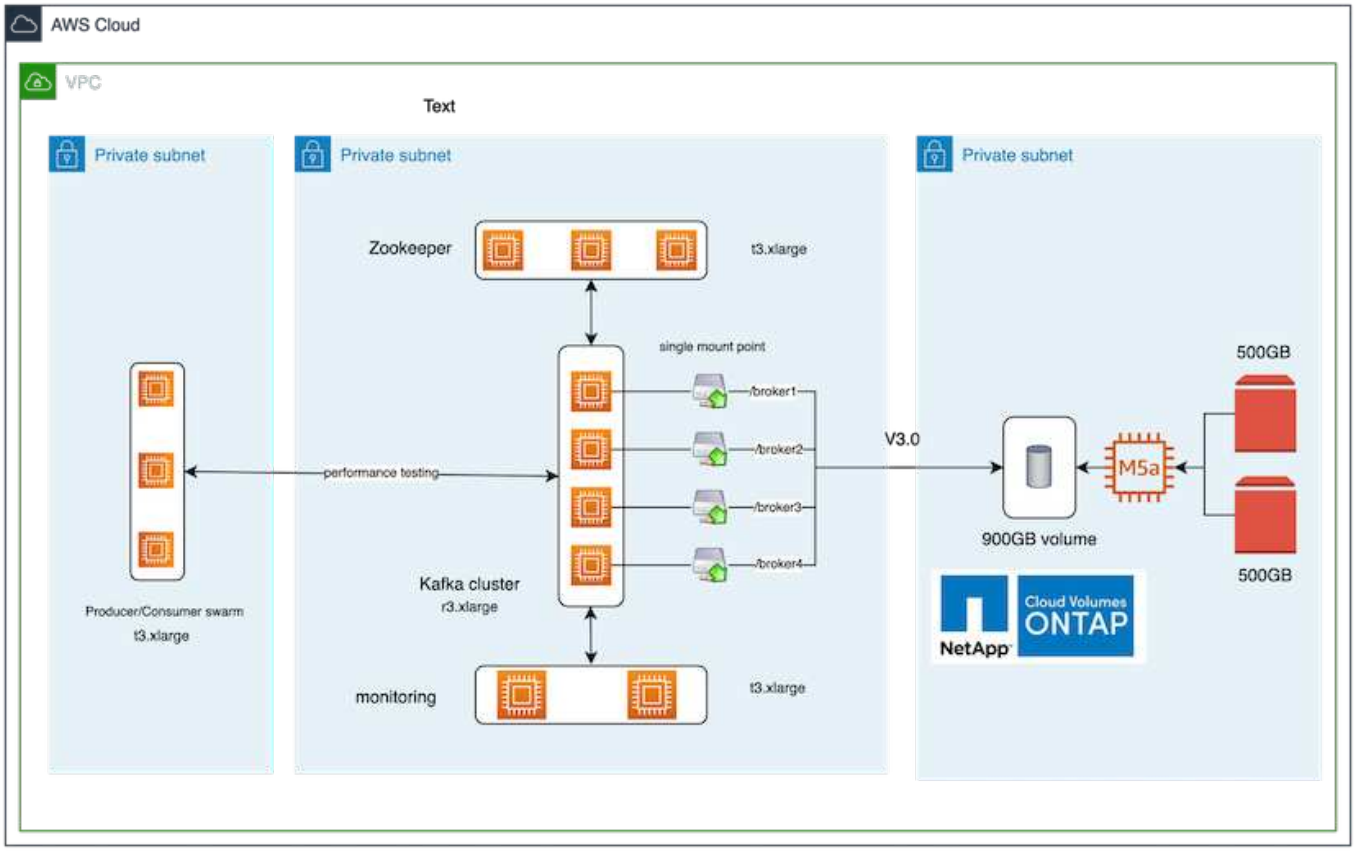
Para la validación funcional, demostramos que un clúster Kafka con un montaje NFSv3 para el almacenamiento no realiza operaciones Kafka como la redistribución de particiones, mientras que otro clúster montado en NFSv4 con la corrección puede realizar las mismas operaciones sin interrupciones.

Configuración de validación

La configuración se ejecuta en AWS. La siguiente tabla muestra los diferentes componentes de la plataforma y la configuración del entorno utilizados para la validación.

Componente de plataforma	Configuración del entorno
Plataforma Confluente, versión 7.2.1	• 3 zookeepers – t3.xlarge • 4 servidores de broker - r3.xlarge • 1 x Grafana – t3.xlarge • 1 centro de control – t3.xlarge • 3 x productor/consumidor
Sistema operativo en todos los nodos	RHEL8.7 o posterior
Instancia de Cloud Volumes ONTAP de NetApp	Instancia de un solo nodo: M5.2xLarge

En la siguiente figura, se muestra la configuración de la arquitectura para esta solución.



Flujo arquitectónico

- **Compute.** utilizamos un clúster Kafka de cuatro nodos con un conjunto de zoomkeeper de tres nodos que se ejecuta en servidores dedicados.
- **Supervisión.** utilizamos dos nodos para una combinación Prometheus-Grafana.
- **Carga de trabajo.** para generar cargas de trabajo, utilizamos un clúster de tres nodos separado que puede producir y consumir desde este clúster Kafka.
- **Almacenamiento.** utilizamos una instancia NetApp Cloud Volumes ONTAP de un solo nodo con dos volúmenes AWS-EBS de 500 GB conectados a la instancia. Estos volúmenes se expusieron al clúster Kafka como volumen NFSv4.1 único mediante una LIF.

Se seleccionaron las propiedades predeterminadas de Kafka para todos los servidores. Lo mismo se hizo para el zookeeper enjambre.

Metodología de las pruebas

1. Actualizar `-is-preserve-unlink-enabled true` al volumen de kafka, como sigue:

```
aws-shantanclastrecall-aws::*> volume create -vserver kafka_svm -volume
kafka_fg_vol01 -aggregate kafka_aggr -size 3500GB -state online -policy
kafka_policy -security-style unix -unix-permissions 0777 -junction-path
/kafka_fg_vol01 -type RW -is-preserve-unlink-enabled true
[Job 32] Job succeeded: Successful
```

2. Se crearon dos clústeres Kafka similares con la siguiente diferencia:
   - **Cluster 1.** el servidor NFS v4.1 back-end con ONTAP versión 9.12.1 listo para producción estaba alojado en una instancia de CVO de NetApp. RHEL 8.7/RHEL 9.1 se han instalado en los intermediarios.
   - **Cluster 2.** el servidor NFS backend era un servidor Linux NFSv3 genérico creado manualmente.
3. Se creó un tema de demostración en los dos clusters de Kafka.

   Clúster 1:

```
[root@ip-172-30-0-160 demo]# kafka-topics --bootstrap-server=172.30.0.160:9092,172.30.0.172:9092,172.30.0.188:90
92,172.30.0.123:9092 --describe --topic __a_demo_topic
Topic: __a_demo_topic   TopicId: 2ty29xfhQLq65HKsUQv-pg PartitionCount: 4       ReplicationFactor: 2    Configs:
 min.insync.replicas=1,segment.bytes=1073741824
        Topic: __a_demo_topic   Partition: 0    Leader: 4       Replicas: 4,1   Isr: 4,1        Offline:
        Topic: __a_demo_topic   Partition: 1    Leader: 2       Replicas: 2,4   Isr: 2,4        Offline:
        Topic: __a_demo_topic   Partition: 2    Leader: 3       Replicas: 3,2   Isr: 3,2        Offline:
        Topic: __a_demo_topic   Partition: 3    Leader: 1       Replicas: 1,3   Isr: 1,3        Offline:
```

   Clúster 2:

```
[root@ip-172-30-0-198 demo]# kafka-topics --bootstrap-server=172.30.0.198:9092,172.30.0.163:9092,172.30.0.221:90
92,172.30.0.204:9092 --describe --topic __a_demo_topic
Topic: __a_demo_topic   TopicId: AwQpsZTQShyeMIhaquCG3Q PartitionCount: 4       ReplicationFactor: 2    Configs:
 min.insync.replicas=1,segment.bytes=1073741824
        Topic: __a_demo_topic   Partition: 0    Leader: 2       Replicas: 2,3   Isr: 2,3        Offline:
        Topic: __a_demo_topic   Partition: 1    Leader: 3       Replicas: 3,1   Isr: 3,1        Offline:
        Topic: __a_demo_topic   Partition: 2    Leader: 1       Replicas: 1,4   Isr: 1,4        Offline:
        Topic: __a_demo_topic   Partition: 3    Leader: 4       Replicas: 4,2   Isr: 4,2        Offline:
```

4. Los datos se han cargado en estos temas recién creados para ambos clústeres. Esto se hizo con el kit de herramientas de prueba de rendimiento-productor que se incluye en el paquete Kafka predeterminado:

```
./kafka-producer-perf-test.sh --topic __a_demo_topic --throughput -1
--num-records 3000000 --record-size 1024 --producer-props acks=all
bootstrap.servers=172.30.0.160:9092,172.30.0.172:9092,172.30.0.188:9092,
172.30.0.123:9092
```

5. Se realizó una comprobación de estado para broker-1 para cada uno de los clústeres mediante telnet:
   - telnet `172.30.0.160 9092`
   - telnet `172.30.0.198 9092`

     En la siguiente captura de pantalla se muestra una comprobación del estado correcta de los agentes en ambos clusters:

```
shantanu@shantanc-mac-0 ~ % telnet 172.30.0.160 9092
Trying 172.30.0.160...
Connected to 172.30.0.160.
Escape character is '^]'.
^[

Connection closed by foreign host.
shantanu@shantanc-mac-0 ~ % telnet 172.30.0.198 9092
Trying 172.30.0.198...
Connected to 172.30.0.198.
Escape character is '^]'.
^[
```

6. Para activar la condición de fallo que provoca que los clústeres de Kafka utilicen volúmenes de almacenamiento NFSv3 se bloquee, hemos iniciado el proceso de reasignación de partición en ambos clústeres. La reasignación de particiones se ha realizado utilizando `kafka-reassign-partitions.sh`. El proceso detallado es el siguiente:
   a. Para reasignar las particiones de un tema de un clúster Kafka, generamos la configuración JSON de reasignación propuesta (esto se realizó para ambos clústeres).

      ```
      kafka-reassign-partitions --bootstrap
      -server=172.30.0.160:9092,172.30.0.172:9092,172.30.0.188:9092,172.30.
      0.123:9092 --broker-list "1,2,3,4" --topics-to-move-json-file
      /tmp/topics.json --generate
      ```

   b. La reasignación JSON generada se guardó en `/tmp/reassignment- file.json`.
   c. El proceso real de reasignación de particiones se ha activado mediante el siguiente comando:

      ```
      kafka-reassign-partitions --bootstrap
      -server=172.30.0.198:9092,172.30.0.163:9092,172.30.0.221:9092,172.30.
      0.204:9092 --reassignment-json-file /tmp/reassignment-file.json
      -execute
      ```

7. Después de unos minutos cuando se completó la reasignación, otra comprobación del estado de los agentes mostró que el clúster con volúmenes de almacenamiento NFSv3 se había ejecutado en un problema tonto de cambio y se había bloqueado, mientras que el clúster 1 con los volúmenes de almacenamiento NFSv4.1 de ONTAP de NetApp con la corrección continuó las operaciones sin ninguna interrupción.

```
shantanu@shantanc-mac-0 ~ % telnet 172.30.0.160 9092
Trying 172.30.0.160...
Connected to 172.30.0.160.
Escape character is '^]'.
^[

Connection closed by foreign host.
shantanu@shantanc-mac-0 ~ % telnet 172.30.0.198 9092
Trying 172.30.0.198...
telnet: connect to address 172.30.0.198: Connection refused
telnet: Unable to connect to remote host
```

- Cluster1-Broker-1 está vivo.
- Cluster2-broker-1 está muerto.

8. Tras comprobar los directorios de registro de Kafka, estaba claro que el clúster 1 con los volúmenes de almacenamiento NFSv4.1 de ONTAP de NetApp con la solución tenía una asignación de particiones limpia, mientras que el clúster 2 con almacenamiento NFSv3 genérico no se debía a problemas tontos de cambio de nombre, lo que provocó el bloqueo. En la siguiente imagen, se muestra el reequilibrio de particiones del clúster 2, lo que dio lugar a un problema de cambio de nombre tonto en el almacenamiento NFSv3.

```
/demo/broker_demo_1/__a_demo_topic-1.b31a8dd60fd443b283ffda2ecca9c2b9-delete:
total 40
drwxr-xr-x.   2 nobody nobody  4096 Sep 19 10:37 .
drwxr-xr-x. 246 nobody nobody 32768 Sep 19 10:36 ..
-rw-r--r--.   1 nobody nobody     5 Sep 19 10:22 .nfs0000000025f9008400000045
-rw-r--r--.   1 nobody nobody     0 Sep 19 10:25 .nfs0000000025f91d6800000048

/demo/broker_demo_1/__a_demo_topic-2:
total 832592
drwxr-xr-x.   2 nobody nobody      4096 Sep 19 10:26 .
drwxr-xr-x. 246 nobody nobody     32768 Sep 19 10:36 ..
-rw-r--r--.   1 nobody nobody         5 Sep 19 10:22 .nfs0000000025f91d5500000046
-rw-r--r--.   1 nobody nobody         0 Sep 19 10:25 .nfs0000000025f91fce00000047
-rw-r--r--.   1 nobody nobody  10485760 Sep 19 10:24 00000000000000000000.index
-rw-r--r--.   1 nobody nobody 848113134 Sep 19 10:24 00000000000000000000.log
-rw-r--r--.   1 nobody nobody  10485756 Sep 19 10:24 00000000000000000000.timeindex
-rw-r--r--.   1 nobody nobody         0 Sep 19 10:16 leader-epoch-checkpoint
-rw-r--r--.   1 nobody nobody        43 Sep 19 10:16 partition.metadata
```

La siguiente imagen muestra un reequilibrado de particiones limpio del clúster 1 mediante NFSv4.1 de NetApp.

```
/demo/broker_demo_1/__a_demo_topic-0:
total 710932
drwxr-xr-x.  2 nobody nobody      4096 Sep 19 10:26 .
drwxr-xr-x. 85 nobody nobody      8192 Sep 19 10:37 ..
-rw-r--r--.  1 nobody nobody  10485760 Sep 19 10:25 00000000000000000000.index
-rw-r--r--.  1 nobody nobody 724167522 Sep 19 10:25 00000000000000000000.log
-rw-r--r--.  1 nobody nobody  10485756 Sep 19 10:25 00000000000000000000.timeindex
-rw-r--r--.  1 nobody nobody         0 Sep 19 10:15 leader-epoch-checkpoint
-rw-r--r--.  1 nobody nobody        43 Sep 19 10:15 partition.metadata

/demo/broker_demo_1/__a_demo_topic-2:
total 780016
drwxr-xr-x.  2 nobody nobody      4096 Sep 19 10:35 .
drwxr-xr-x. 85 nobody nobody      8192 Sep 19 10:37 ..
-rw-r--r--.  1 nobody nobody  10485760 Sep 19 10:36 00000000000000000000.index
-rw-r--r--.  1 nobody nobody 794575786 Sep 19 10:36 00000000000000000000.log
-rw-r--r--.  1 nobody nobody  10485756 Sep 19 10:36 00000000000000000000.timeindex
-rw-r--r--.  1 nobody nobody         0 Sep 19 10:35 leader-epoch-checkpoint
-rw-r--r--.  1 nobody nobody        43 Sep 19 10:35 partition.metadata
```

¿Por qué NFS de NetApp para las cargas de trabajo de Kafka?

Ahora que hay una solución para el tonto problema de cambio de nombre del almacenamiento NFS con Kafka, se pueden crear puestas en marcha sólidas que aprovechan el almacenamiento ONTAP de NetApp para su carga de trabajo Kafka. Esto no solo reduce significativamente los gastos operativos, sino que también aporta las siguientes ventajas a los clústeres de Kafka:

- * Reducción del uso de la CPU en los intermediarios de Kafka.* utilizando el almacenamiento desagregado de ONTAP de NetApp separa las operaciones de I/o de disco del intermediario y, por tanto, reduce el espacio físico utilizado de la CPU.
- **Tiempo de recuperación de broker más rápido.** desde que el almacenamiento desagregado de ONTAP de NetApp se comparte en los nodos de broker de Kafka, una nueva instancia informática puede sustituir a un intermediario defectuoso en cualquier momento, en comparación con las puestas en marcha convencionales de Kafka sin tener que volver a crear los datos.
- * Eficiencia del almacenamiento.* como la capa de almacenamiento de la aplicación se aprovisiona ahora a través de ONTAP de NetApp, los clientes pueden aprovechar las ventajas de la eficiencia del almacenamiento que incluye ONTAP, como la compresión de datos inline, la deduplicación y la compactación.

Estas ventajas se probaron y validaron en casos de prueba que comentamos detalladamente en esta sección.

Reducción del uso de CPU en Kafka Broker

Descubrimos que el aprovechamiento de la CPU general es inferior al de su homólogo de DAS, cuando ejecutamos cargas de trabajo similares en dos clústeres de Spermiate Kafka que eran idénticas en sus especificaciones técnicas, pero que diferían en sus tecnologías de almacenamiento. El uso general de la CPU no sólo es inferior cuando el clúster Kafka utiliza almacenamiento ONTAP, sino que, además, el aumento del uso de la CPU ha mostrado un gradiente más suave que en un clúster Kafka basado en DAS.

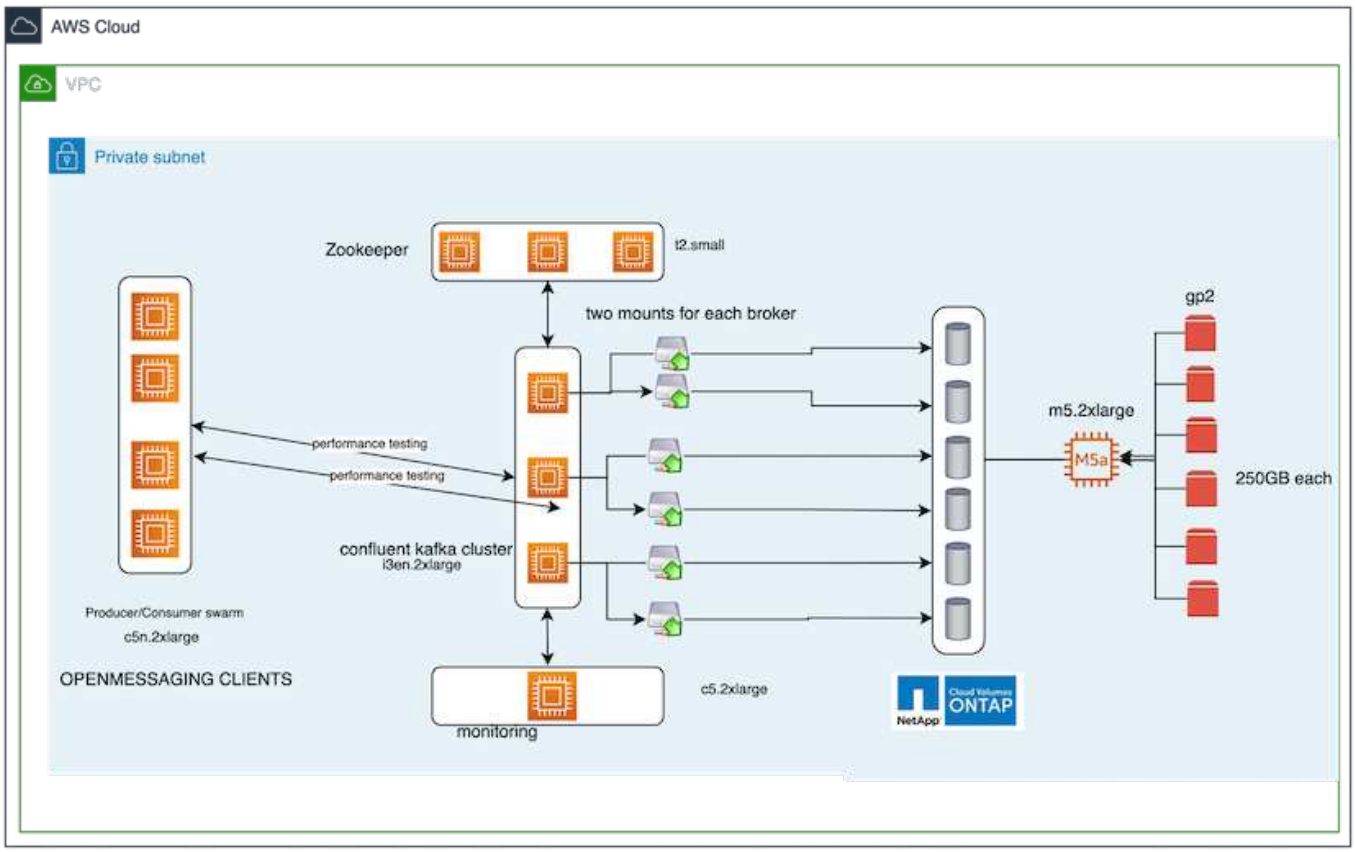
Configuración de la arquitectura

La siguiente tabla muestra la configuración del entorno utilizada para demostrar la reducción del uso de CPU.

Componente de plataforma	Configuración del entorno
Kafka 3.2.3 herramienta de Benchmarking: OpenMessaging	• 3 zookeepers – t2.pequeño • 3 servidores de broker: i3en.2xlarge • 1 x Grafana – c5n.2xgrande • 4 x productor/consumidor — c5n.2xgrande
Sistema operativo en todos los nodos	RHEL 8.7 o posterior
Instancia de Cloud Volumes ONTAP de NetApp	Instancia de un solo nodo: M5.2xLarge

Herramienta de evaluación comparativa

La herramienta de evaluación comparativa utilizada en este caso de prueba es la "Mensajería abierta" marco. OpenMessaging es independiente del lenguaje y está neutral en todos los proveedores; proporciona directrices del sector para finanzas, comercio electrónico, Internet de las cosas y Big Data; además, ayuda a desarrollar aplicaciones de mensajería y transmisión de datos en sistemas y plataformas heterogéneos. La figura siguiente muestra la interacción de los clientes de OpenMessaging con un clúster Kafka.



- **Compute.** utilizamos un clúster Kafka de tres nodos con un conjunto de zoomkeeper de tres nodos que se ejecuta en servidores dedicados. Cada agente tenía dos puntos de montaje de NFSv4.1 en un único volumen de la instancia de CVO de NetApp a través de un LIF dedicado.

- **Supervisión.** utilizamos dos nodos para una combinación Prometheus-Grafana. Para generar cargas de trabajo, tenemos un clúster de tres nodos separado que puede producir y consumir a partir de este clúster Kafka.
- **Almacenamiento.** utilizamos una instancia Cloud Volumes ONTAP de NetApp de un solo nodo con seis volúmenes AWS-EBS de 250 GB montados en la instancia. Estos volúmenes se expusieron entonces al clúster Kafka como seis volúmenes de NFSv4.1 mediante LIF dedicadas.
- **Configuración.** los dos elementos configurables en este caso de prueba fueron los agentes Kafka y las cargas de trabajo de OpenMessaging.
    - **Broker config.** las siguientes especificaciones fueron seleccionadas para los corredores Kafka. Utilizamos el factor de replicación 3 para todas las mediciones, tal y como se destaca a continuación.

```
broker.id=1
advertised.listeners=PLAINTEXT://172.30.0.185:9092
log.dirs=/mnt/data-1
zookeeper.connect=172.30.0.13:2181,172.30.0.108:2181,172.30.0.253:2181
num.replica.fetchers=8
message.max.bytes=10485760
replica.fetch.max.bytes=10485760
num.network.threads=8
default.replication.factor=3
replica.lag.time.max.ms=100000000
replica.fetch.max.bytes=1048576
replica.fetch.wait.max.ms=500
num.replica.fetchers=1
replica.high.watermark.checkpoint.interval.ms=5000
fetch.purgatory.purge.interval.requests=1000
producer.purgatory.purge.interval.requests=1000
replica.socket.timeout.ms=30000
replica.socket.receive.buffer.bytes=65536
```

- **Configuración de carga de trabajo de OpenMessaging Benchmark (OMB).** se proporcionaron las siguientes especificaciones. Hemos especificado una tasa de producción objetivo, que se destaca a continuación.

```
name: 4 producer / 4 consumers on 1 topic
topics: 1
partitionsPerTopic: 100
messageSize: 1024
payloadFile: "payload/payload-1Kb.data"
subscriptionsPerTopic: 1
consumerPerSubscription: 4
producersPerTopic: 4
producerRate: 40000
consumerBacklogSizeGB: 0
testDurationMinutes: 5
```

Metodología de las pruebas

1. Se crearon dos grupos similares, cada uno con su propio conjunto de enjambres de racimo de benchmarking.
   - **Cluster 1.** clúster Kafka basado en NFS.
   - **Cluster 2.** clúster Kafka basado en DAS.
2. Con un comando OpenMessaging, se activaron cargas de trabajo similares en cada clúster.

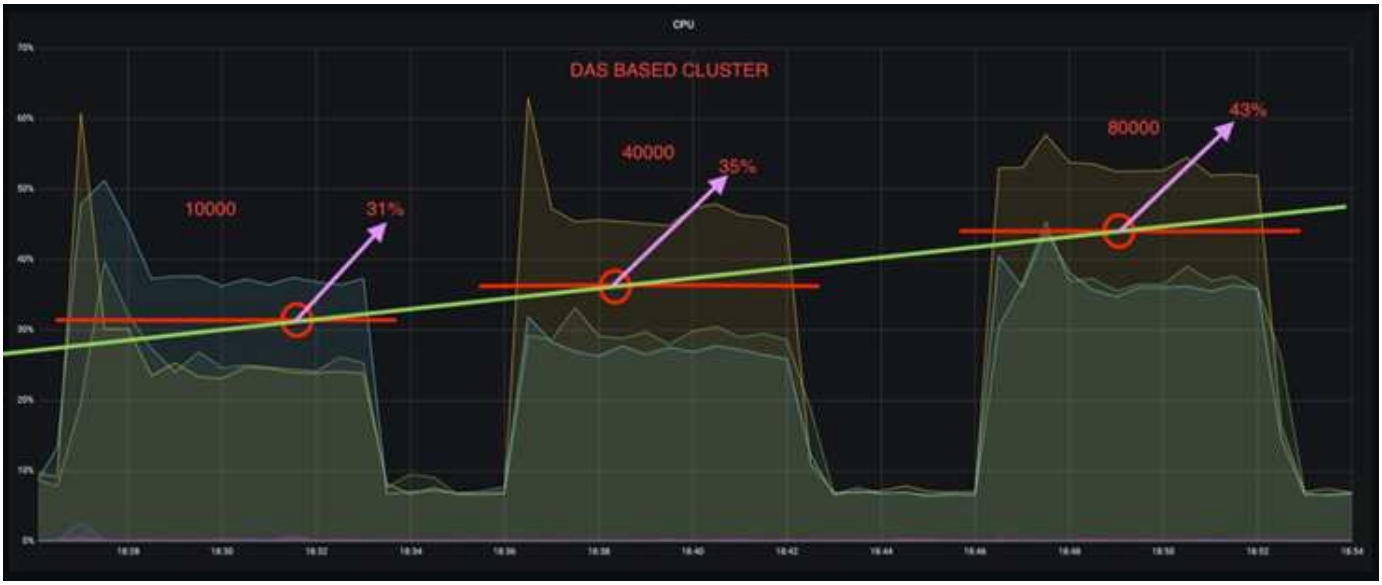
   ```
   sudo bin/benchmark --drivers driver-kafka/kafka-group-all.yaml
   workloads/1-topic-100-partitions-1kb.yaml
   ```

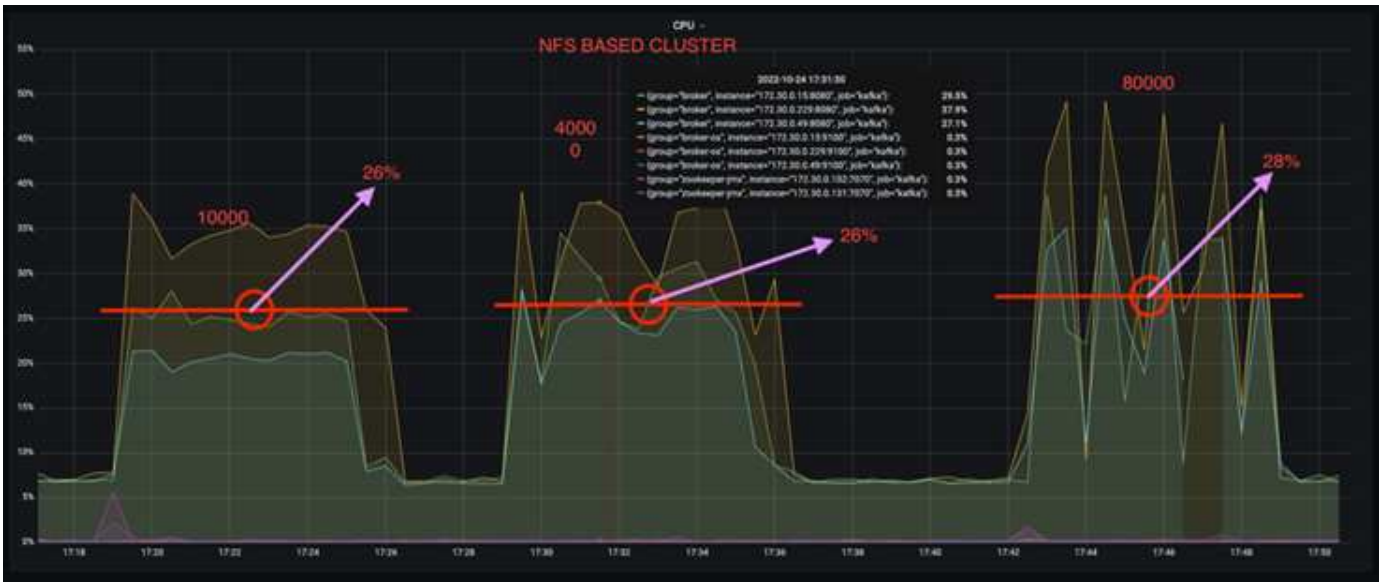
3. La configuración de la tasa de producción se aumentó en cuatro iteraciones, y se registró un aprovechamiento de la CPU en Grafana. La tasa de producción se ha establecido en los siguientes niveles:
   - 10,000
   - 40,000
   - 80,000
   - 100,000

Observación

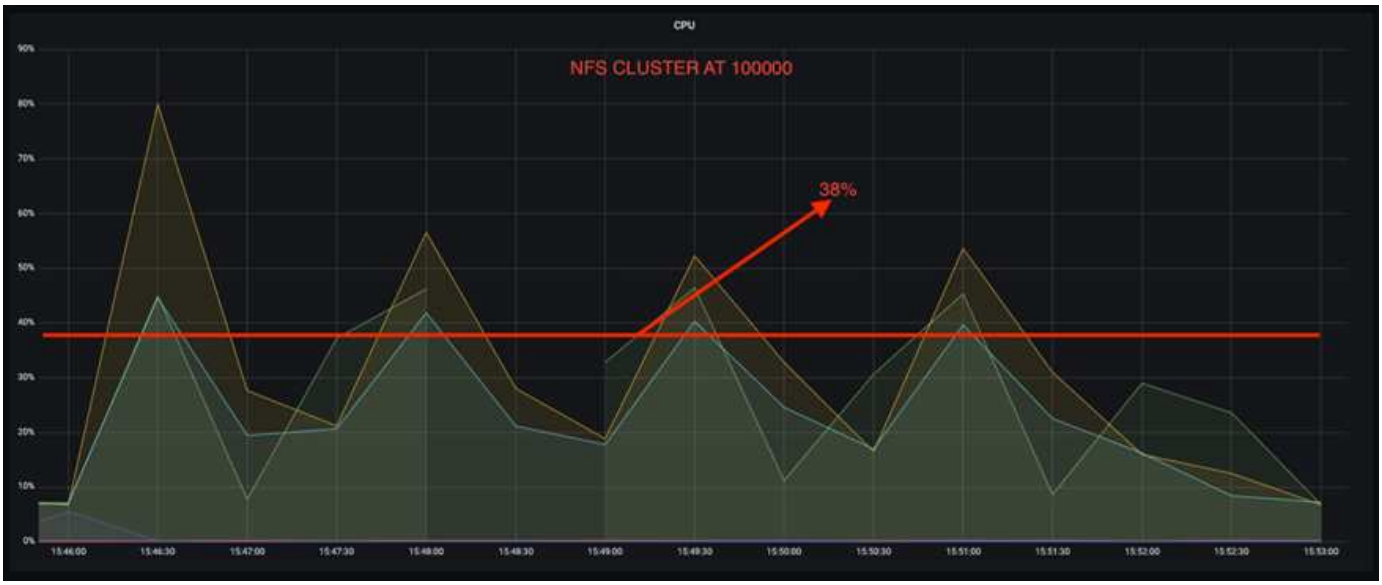
Se obtienen dos principales ventajas de usar el almacenamiento NFS de NetApp con Kafka:

- **Puede reducir el uso de la CPU en casi un tercio.** el uso general de la CPU en cargas de trabajo similares fue menor para NFS en comparación con los SSD DAS; los ahorros varían de un 5% para tasas de producción más bajas a un 32% para tasas de producción más altas.
- * Una reducción de tres veces en la deriva de la utilización de la CPU a tasas de producción más altas.* como se esperaba, hubo una deriva ascendente para el aumento de la utilización de la CPU a medida que se aumentaron las tasas de producción. Sin embargo, el uso de la CPU en los agentes Kafka que utilizan DAS ha aumentado del 31% con la tasa de producción inferior al 70% con la tasa de producción más alta, lo que representa un aumento del 39%. Sin embargo, con un back-end de almacenamiento NFS, el uso de CPU ha aumentado del 26 % al 38 %, lo que representa un aumento del 12 %.





Asimismo, en 100,000 mensajes, el almacenamiento DAS muestra más uso de CPU que un clúster NFS.



Recuperación de agentes más rápida

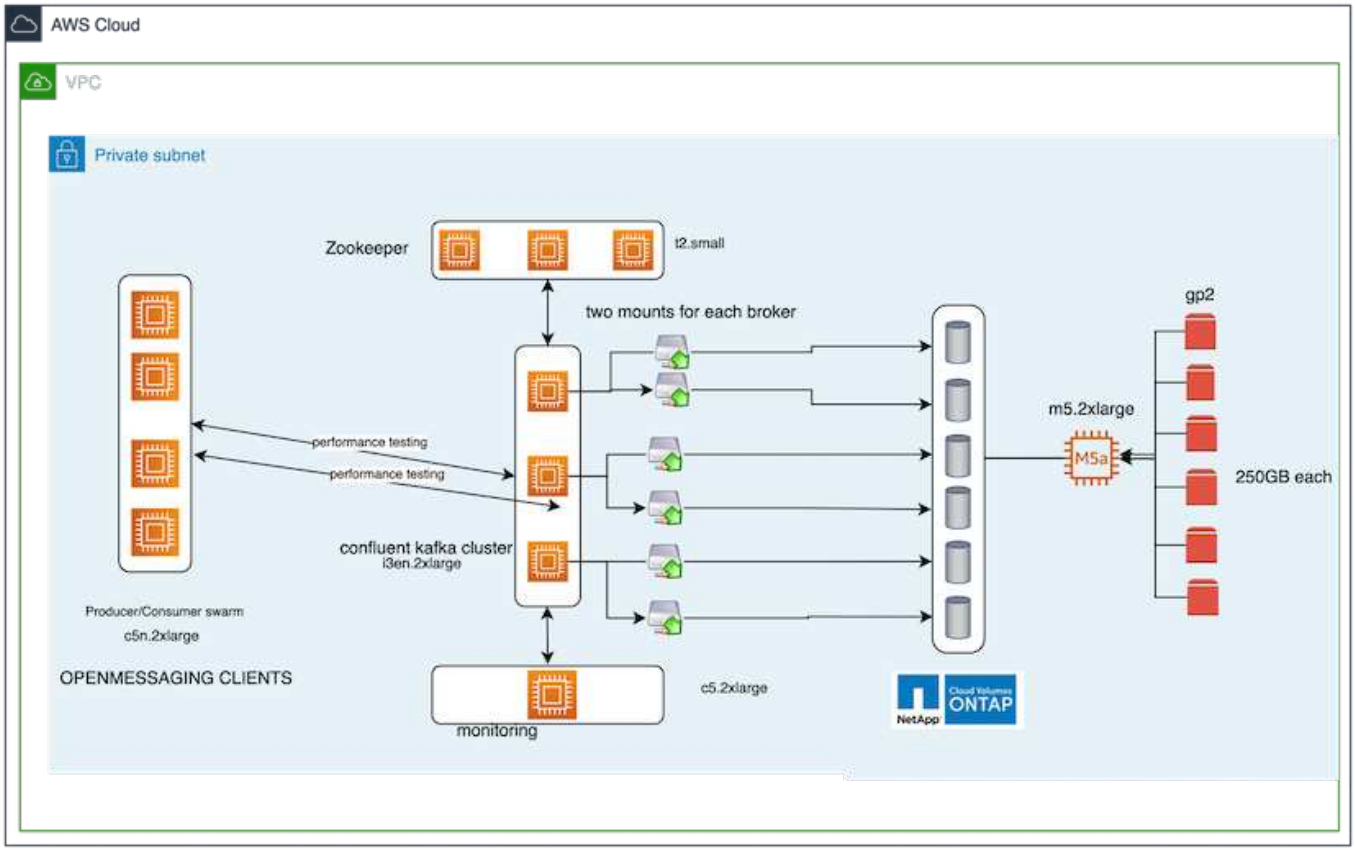
Descubrimos que los agentes de Kafka se recuperan con mayor rapidez cuando se utiliza el almacenamiento NFS compartido de NetApp. Cuando un agente se bloquea en un clúster de Kafka, este agente se puede reemplazar por un agente en buen estado con un mismo ID de agente. Tras realizar este caso de prueba, descubrimos que, en el caso de un clúster Kafka basado en DAS, el clúster recompila los datos en un nuevo agente de buena salud añadido, lo cual requiere mucho tiempo. En el caso de un clúster Kafka basado en NFS de NetApp, el agente de sustitución sigue leyendo datos del directorio de registros anterior y recupera mucho más rápido.

Configuración de la arquitectura

En la siguiente tabla se muestra la configuración del entorno de un clúster de Kafka con NAS.

Componente de plataforma	Configuración del entorno
Kafka 3.2.3	• 3 zookeepers – t2.pequeño • 3 servidores de broker: i3en.2xlarge • 1 x Grafana – c5n.2xgrande • 4 x productor/consumidor — c5n.2xgrande • 1 nodo Kafka de backup: I3en.2xgrande
Sistema operativo en todos los nodos	RHEL8.7 o posterior
Instancia de Cloud Volumes ONTAP de NetApp	Instancia de un solo nodo: M5.2xLarge

En la figura siguiente se muestra la arquitectura de un clúster Kafka basado en NAS.



- **Compute.** un clúster Kafka de tres nodos con un conjunto de zomantenimiento de tres nodos que se ejecuta en servidores dedicados. Cada agente tiene dos puntos de montaje NFS en un único volumen en la instancia de NetApp CVO a través de un LIF dedicado.
- **Supervisión.** dos nodos para una combinación Prometheus-Grafana. Para generar cargas de trabajo, utilizamos un clúster de tres nodos independiente que puede producir y consumir con este clúster de Kafka.
- **Almacenamiento.** una instancia de NetApp Cloud Volumes ONTAP de un solo nodo con seis volúmenes AWS-EBS de 250 GB montados en la instancia. A continuación, estos volúmenes se exponen al clúster de Kafka en seis volúmenes NFS mediante LIF dedicadas.
- **Configuración de Broker.** el único elemento configurable en este caso de prueba son agentes Kafka. Se seleccionaron las siguientes especificaciones para los corredores Kafka. La `replica.lag.time.mx.ms` Se establece en un valor alto porque determina la rapidez con la que se sale un nodo concreto de la lista ISR. Cuando cambia entre nodos defectuosos y sanos, no desea que ese ID de broker se excluya de la lista ISR.

```
broker.id=1
advertised.listeners=PLAINTEXT://172.30.0.185:9092
log.dirs=/mnt/data-1
zookeeper.connect=172.30.0.13:2181,172.30.0.108:2181,172.30.0.253:2181
num.replica.fetchers=8
message.max.bytes=10485760
replica.fetch.max.bytes=10485760
num.network.threads=8
default.replication.factor=3
replica.lag.time.max.ms=100000000
replica.fetch.max.bytes=1048576
replica.fetch.wait.max.ms=500
num.replica.fetchers=1
replica.high.watermark.checkpoint.interval.ms=5000
fetch.purgatory.purge.interval.requests=1000
producer.purgatory.purge.interval.requests=1000
replica.socket.timeout.ms=30000
replica.socket.receive.buffer.bytes=65536
```

Metodología de las pruebas

1. Se crearon dos clústeres similares:
   - Un clúster fluido basado en EC2.
   - Un clúster fluido basado en NFS de NetApp.
2. Se creó un nodo Kafka en espera con una configuración idéntica a los nodos del clúster Kafka original.
3. En cada uno de los clústeres se creó un tema de ejemplo y se rellenaron aproximadamente 110 GB de datos en cada uno de los agentes de valores.
   - **Clúster basado en EC2.** se asigna Un directorio de datos de Kafka broker `/mnt/data-2` (En la siguiente figura, Broker-1 de cluster1 [terminal izquierdo]).
   - **Clúster basado en NFS de NetApp.** un directorio de datos de Kafka Broker está montado en punto NFS `/mnt/data` (En la siguiente figura, Broker-1 de cluster2 [terminal derecho]).

4. En cada uno de los clústeres, Broker-1 se terminó para activar un proceso de recuperación de broker fallido.
5. Una vez que el broker fue terminado, la dirección IP del broker fue asignada como IP secundaria al broker en espera. Esto fue necesario porque a un corredor de un clúster de Kafka se le identifica lo siguiente:
   - **Dirección IP.** asignado reasignando el IP de broker fallido al intermediario en espera.
   - **ID de broker.** se configuró en el broker en espera `server.properties`.
6. Tras la asignación de IP, el servicio Kafka se inició en el agente de reserva.
7. Tras un tiempo, los registros del servidor se han extraído para comprobar el tiempo que se tarda en crear datos en el nodo de reemplazo del clúster.

Observación

La recuperación de los intermediarios de Kafka fue casi nueve veces más rápida. Se constató que el tiempo que se tardaba en recuperar un nodo de agente fallido era significativamente más rápido cuando se usa el almacenamiento compartido NFS de NetApp en comparación con el uso de SSD DAS en un clúster Kafka. En 1 TB de datos de temas, el tiempo de recuperación de un clúster basado en DAS era de 48 minutos, en comparación con menos de 5 minutos para un clúster Kafka basado en NetApp-NFS.

Observamos que el clúster basado en EC2 tardó 10 minutos en reconstruir los 110 GB de datos en el nuevo nodo de agente, mientras que el clúster basado en NFS completó la recuperación en 3 minutos. También observamos en los registros in que los offsets de los consumidores para las particiones para EC2 eran 0, mientras que, en el clúster NFS, los offsets de los consumidores se recogían del intermediario anterior.

```
[2022-10-31 09:39:17,747] INFO [LogLoader partition=test-topic-51R3EWs-
0000-55, dir=/mnt/kafka-data/broker2] Reloading from producer snapshot and
rebuilding producer state from offset 583999 (kafka.log.UnifiedLog$)
[2022-10-31 08:55:55,170] INFO [LogLoader partition=test-topic-qbVsEZg-
0000-8, dir=/mnt/data-1] Loading producer state till offset 0 with message
format version 2 (kafka.log.UnifiedLog$)
```

Clúster basado en DAS

1. El nodo de backup se inició a las 08:55:53,730.

```
2  [2022-10-31 08:55:53,661] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotia
3  [2022-10-31 08:55:53,727] INFO Registered signal handlers for TERM, INT, HUP (or
4  [2022-10-31 08:55:53,730] INFO starting (kafka.server.KafkaServer)
5  [2022-10-31 08:55:53,730] INFO Connecting to zookeeper on 172.30.0.17:2181,172.3
6  [2022-10-31 08:55:53,755] INFO [ZooKeeperClient Kafka server] Initializing a new
```

2. El proceso de recompilación de datos finalizó a las 09:05:24,860. El procesamiento de 110 GB de datos requería aproximadamente 10 minutos.

```
[2022-10-31 09:05:24,860] INFO [ReplicaFetcherManager on broker 1] Removed fetcher for
partitions HashSet(test-topic-qbVsEZg-0000-95, test-topic-qbVsEZg-0000-5,
test-topic-qbVsEZg-0000-41, test-topic-qbVsEZg-0000-23, test-topic-qbVsEZg-0000-11,
test-topic-qbVsEZg-0000-47, test-topic-qbVsEZg-0000-83, test-topic-qbVsEZg-0000-35,
test-topic-qbVsEZg-0000-89, test-topic-qbVsEZg-0000-71, test-topic-qbVsEZg-0000-53,
test-topic-qbVsEZg-0000-29, test-topic-qbVsEZg-0000-59, test-topic-qbVsEZg-0000-77,
test-topic-qbVsEZg-0000-65, test-topic-qbVsEZg-0000-17)
(kafka.server.ReplicaFetcherManager)
```

Clúster basado en NFS

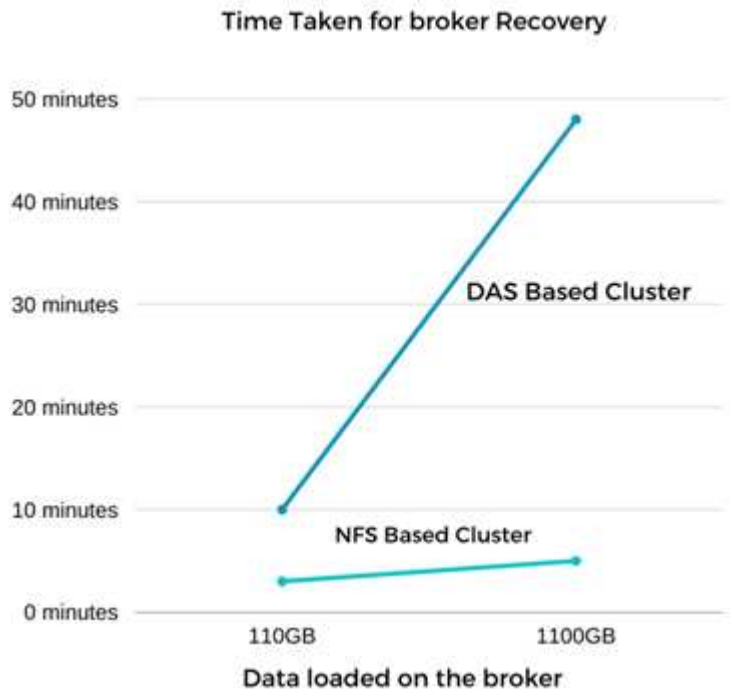
1. El nodo de backup se inició a las 09:39:17,213. A continuación se resalta la entrada del registro inicial.

```
2  [2022-10-31 09:39:17,142] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiati
3  [2022-10-31 09:39:17,211] INFO Registered signal handlers for TERM, INT, HUP (org.
4  [2022-10-31 09:39:17,213] INFO starting (kafka.server.KafkaServer)
5  [2022-10-31 09:39:17,214] INFO Connecting to zookeeper on 172.30.0.22:2181,172.30.
6  [2022-10-31 09:39:17,238] INFO [ZooKeeperClient Kafka server] Initializing a new s
7  [2022-10-31 09:39:17,244] INFO Client environment:zookeeper.version=3.6.3--6401e4a
8  [2022-10-31 09:39:17,244] INFO Client environment:host.name=ip-172-30-0-110.ec2.in
9  [2022-10-31 09:39:17,244] INFO Client environment:java.version=11.0.17 (org.apache
```

2. El proceso de reconstrucción de los datos terminó a las 09:42:29,115. El procesamiento de 110 GB de datos requería aproximadamente 3 minutos.

```
[2022-10-31 09:42:29,115] INFO [GroupMetadataManager brokerId=1] Finished loading offsets
and group metadata from __consumer_offsets-20 in 28478 milliseconds for epoch 3, of which
28478 milliseconds was spent in the scheduler.
(kafka.coordinator.group.GroupMetadataManager)
```

La prueba fue repetida para los agentes que tenían alrededor de 1 TB de datos, lo que supuso aproximadamente 48 minutos para el sistema DAS y 3 minutos para NFS. Los resultados se muestran en el siguiente gráfico.



Eficiencia del almacenamiento

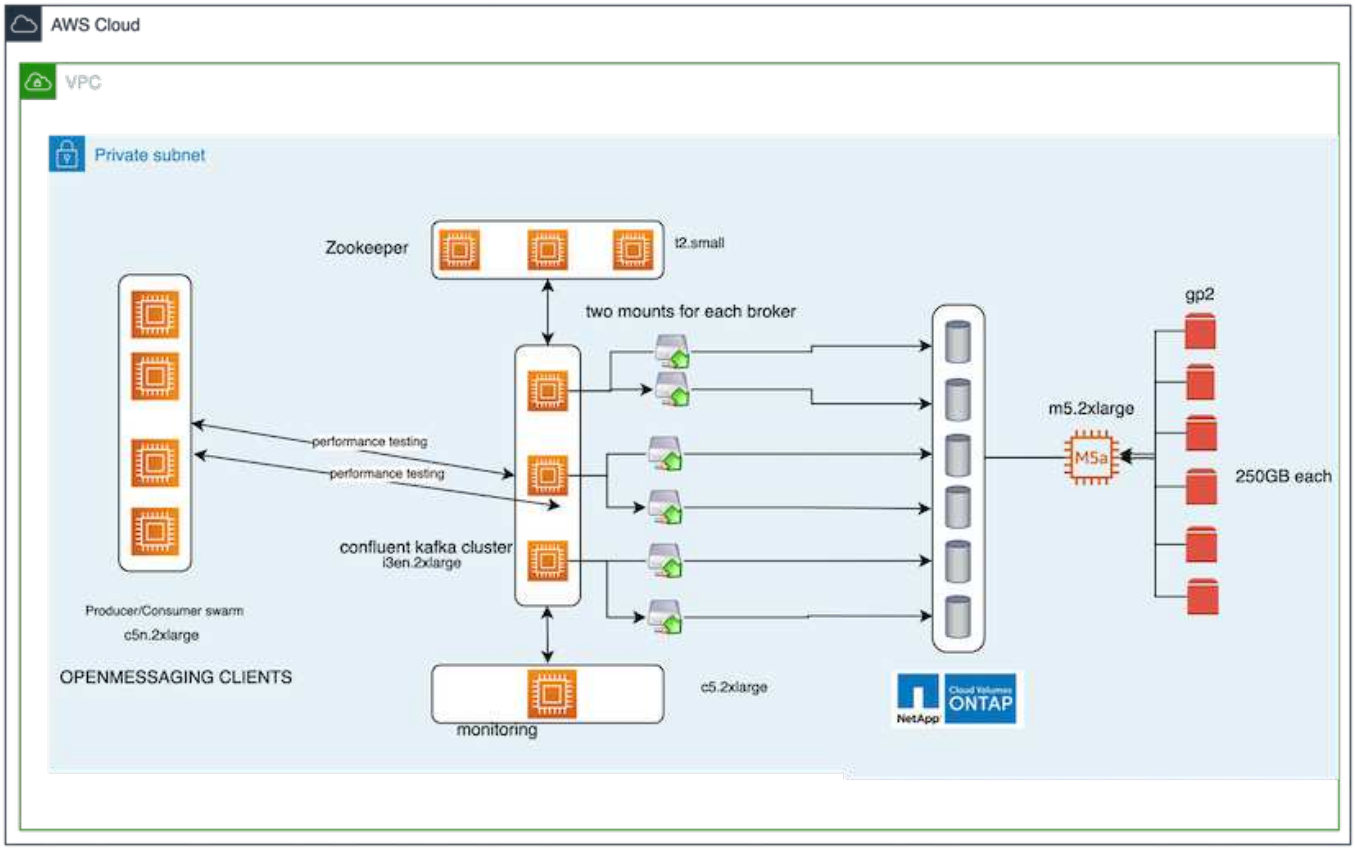
Como la capa de almacenamiento del clúster Kafka se aprovisionaba a través de ONTAP de NetApp, obtuvimos todas las funcionalidades de eficiencia del almacenamiento de ONTAP. Esto se probó generando una cantidad significativa de datos en un clúster de Kafka con almacenamiento NFS aprovisionado en Cloud Volumes ONTAP. Pudimos ver que hubo una reducción significativa del espacio gracias a las funcionalidades de ONTAP.

Configuración de la arquitectura

En la siguiente tabla se muestra la configuración del entorno de un clúster de Kafka con NAS.

Componente de plataforma	Configuración del entorno
Kafka 3.2.3	• 3 zookeepers – t2.pequeño • 3 servidores de broker: i3en.2xlarge • 1 x Grafana – c5n.2xgrande • 4 x productor/consumidor — c5n.2xgrande *
Sistema operativo en todos los nodos	RHEL8.7 o posterior
Instancia de Cloud Volumes ONTAP de NetApp	Instancia de un solo nodo: M5.2xLarge

En la figura siguiente se muestra la arquitectura de un clúster Kafka basado en NAS.



- **Compute.** utilizamos un clúster Kafka de tres nodos con un conjunto de zoomkeeper de tres nodos que se ejecuta en servidores dedicados. Cada agente tenía dos puntos de montaje NFS en un único volumen en la instancia de NetApp CVO a través de un LIF dedicado.
- **Supervisión.** utilizamos dos nodos para una combinación Prometheus-Grafana. Para generar cargas de trabajo, utilizamos un clúster de tres nodos independiente que podía producir y consumir este clúster Kafka.
- **Almacenamiento.** utilizamos una instancia Cloud Volumes ONTAP de NetApp de un solo nodo con seis volúmenes AWS-EBS de 250 GB montados en la instancia. Estos volúmenes se expusieron a continuación al clúster de Kafka en seis volúmenes NFS mediante LIF dedicadas.
- **Configuración.** los elementos configurables en este caso de prueba fueron los agentes Kafka.

La compresión se apagó al final del productor, lo que permitió a los productores generar un alto rendimiento.

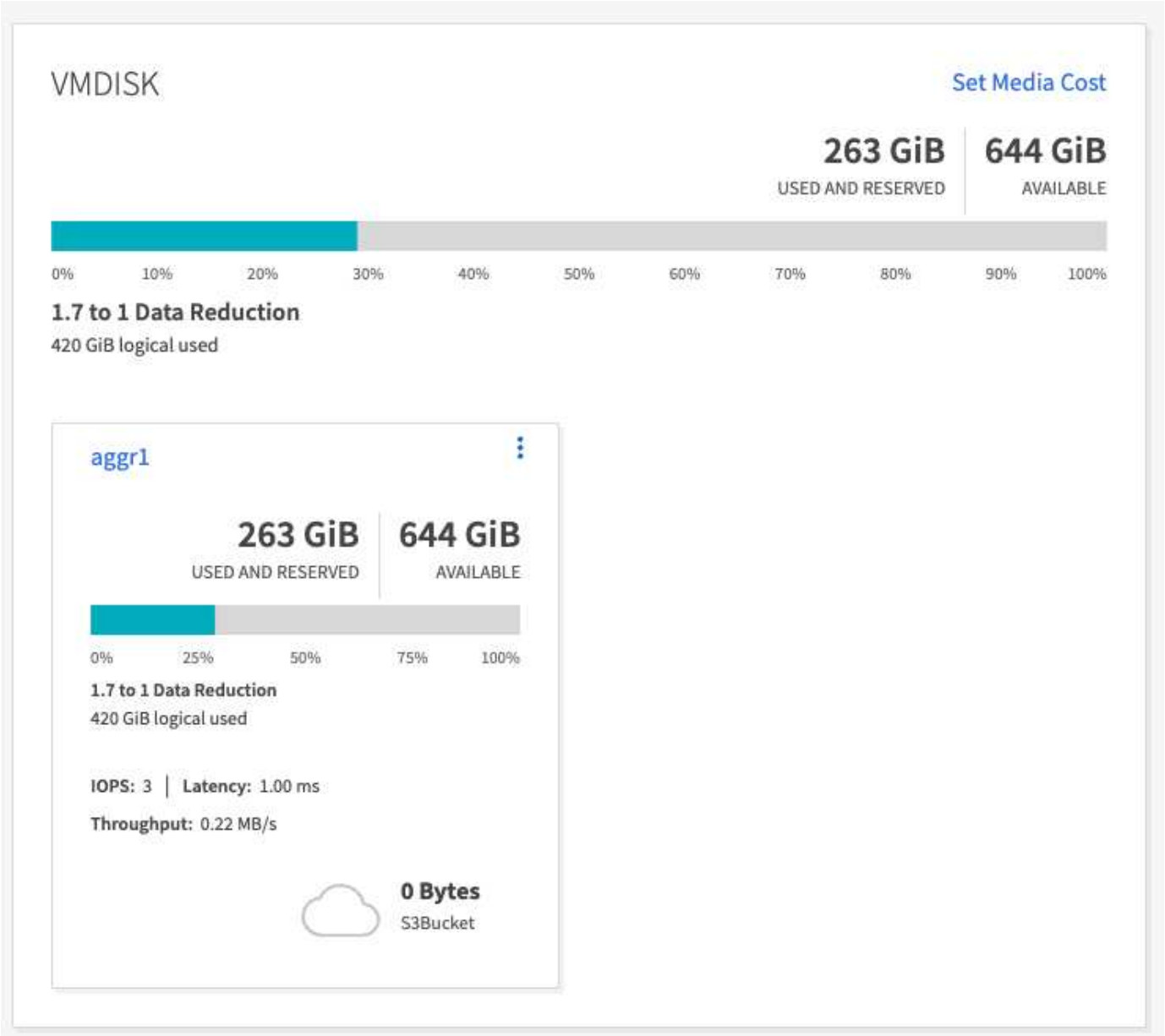
En lugar de eso, la capa informática gestionó la eficiencia del almacenamiento.

Metodología de las pruebas

1. Se aprovisionó un clúster de Kafka con las especificaciones mencionadas anteriormente.
2. En el clúster, se produjeron unos 350 GB de datos con la herramienta de puntos de referencia OpenMessaging.
3. Una vez completada la carga de trabajo, las estadísticas de eficiencia del almacenamiento se recogieron con ONTAP System Manager y CLI.

Observación

Para los datos generados con la herramienta OMB, observamos un ahorro de espacio de ~33 % con una relación de eficiencia de almacenamiento de 1.70:1. Tal y como se aprecia en las siguientes figuras, el espacio lógico utilizado por los datos producidos era de 420,3 GB y el espacio físico utilizado para almacenar los datos era de 281,7 GB.



```
shantanuCVOinstancenew::> df -h -S

Warning: The "-S" parameter is deprecated and may be removed in a future release. To show the efficiency ratio use "aggr show-efficiency"
         command.
Filesystem               used      total-saved    %total-saved    deduplicated    %deduplicated    compressed    %compressed  Vserver
/vol/vol0/             7319MB               0B              0%              0B               0%            0B             0%  shantanuCVOinstancenew-01
/vol/kafka_vol/         281GB            138GB             33%           138GB              33%            0B             0%  svm_shantanuCVOinstancenew
/vol/svm_shantanuCVOinstancenew_root/
                        660KB               0B              0%              0B               0%            0B             0%  svm_shantanuCVOinstancenew
3 entries were displayed.
```

```
                    Name of the Aggregate: aggr1
              Node where Aggregate Resides: shantanuCVOinstancenew-01
             Total Storage Efficiency Ratio: 1.70:1
Total Data Reduction Efficiency Ratio Without Snapshots: 1.70:1
Total Data Reduction Efficiency Ratio without snapshots and flexclones: 1.70:1
         Logical Space Used for All Volumes: 420.3GB
        Physical Space Used for All Volumes: 281.7GB
```

Información general y validación del rendimiento en AWS

Se realizó una prueba de rendimiento de un clúster Kafka con capa de almacenamiento montada en NFS de NetApp en el cloud de AWS. Los ejemplos de pruebas comparativas se describen en las siguientes secciones.

Kafka en el cloud de AWS con Cloud Volumes ONTAP de NetApp (pareja de alta disponibilidad y nodo único)

Se realizó una prueba de rendimiento de un clúster de Kafka con Cloud Volumes ONTAP de NetApp (pareja de alta disponibilidad) en el cloud de AWS. Esta evaluación comparativa se describe en las siguientes secciones.

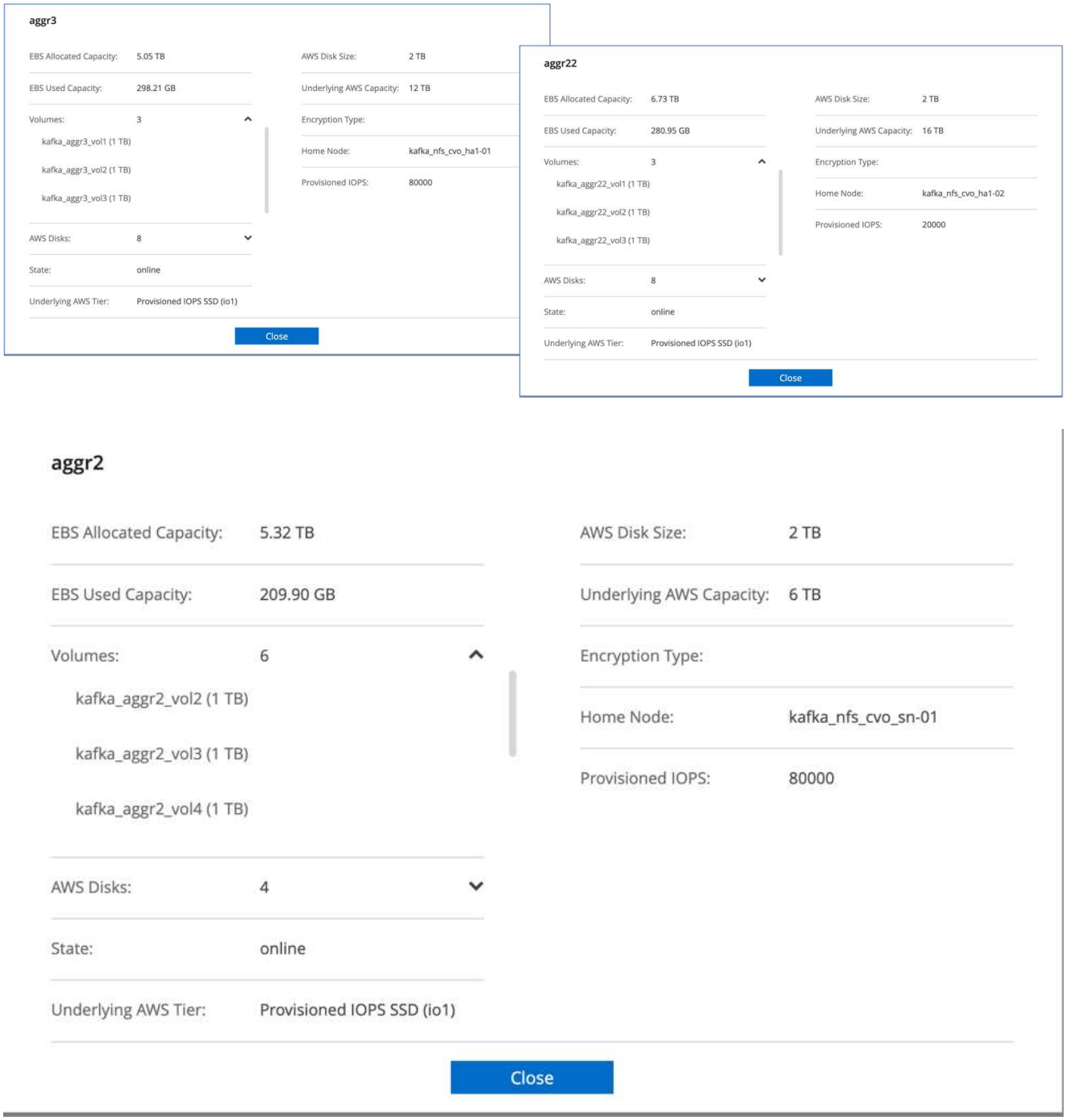
Configuración de la arquitectura

En la siguiente tabla se muestra la configuración del entorno de un clúster de Kafka con NAS.

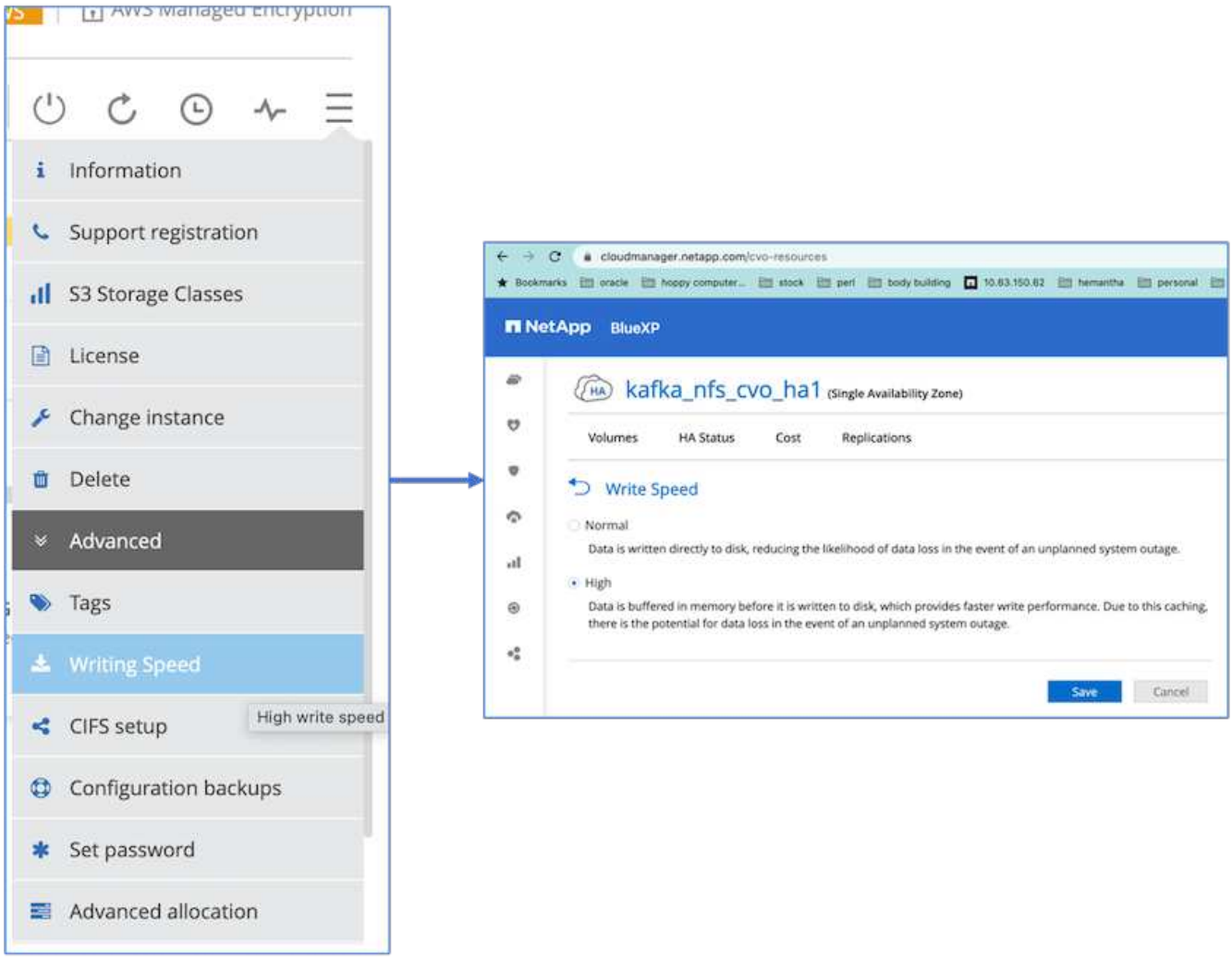
Componente de plataforma	Configuración del entorno
Kafka 3.2.3	• 3 zookeepers – t2.pequeño • 3 servidores de broker: i3en.2xlarge • 1 x Grafana – c5n.2xgrande • 4 x productor/consumidor — c5n.2xgrande *
Sistema operativo en todos los nodos	RHEL8.6
Instancia de Cloud Volumes ONTAP de NetApp	Instancia de par DE ALTA DISPONIBILIDAD – m5dn.12xLarge x 2 node Single Node Instance - m5dn.12xLarge x 1 node

Configuración de ONTAP para volúmenes del clúster de NetApp

1. Para el par de alta disponibilidad de Cloud Volumes ONTAP, hemos creado dos agregados con tres volúmenes en cada agregado de cada controladora de almacenamiento. Para el nodo único de Cloud Volumes ONTAP, creamos seis volúmenes en un agregado.

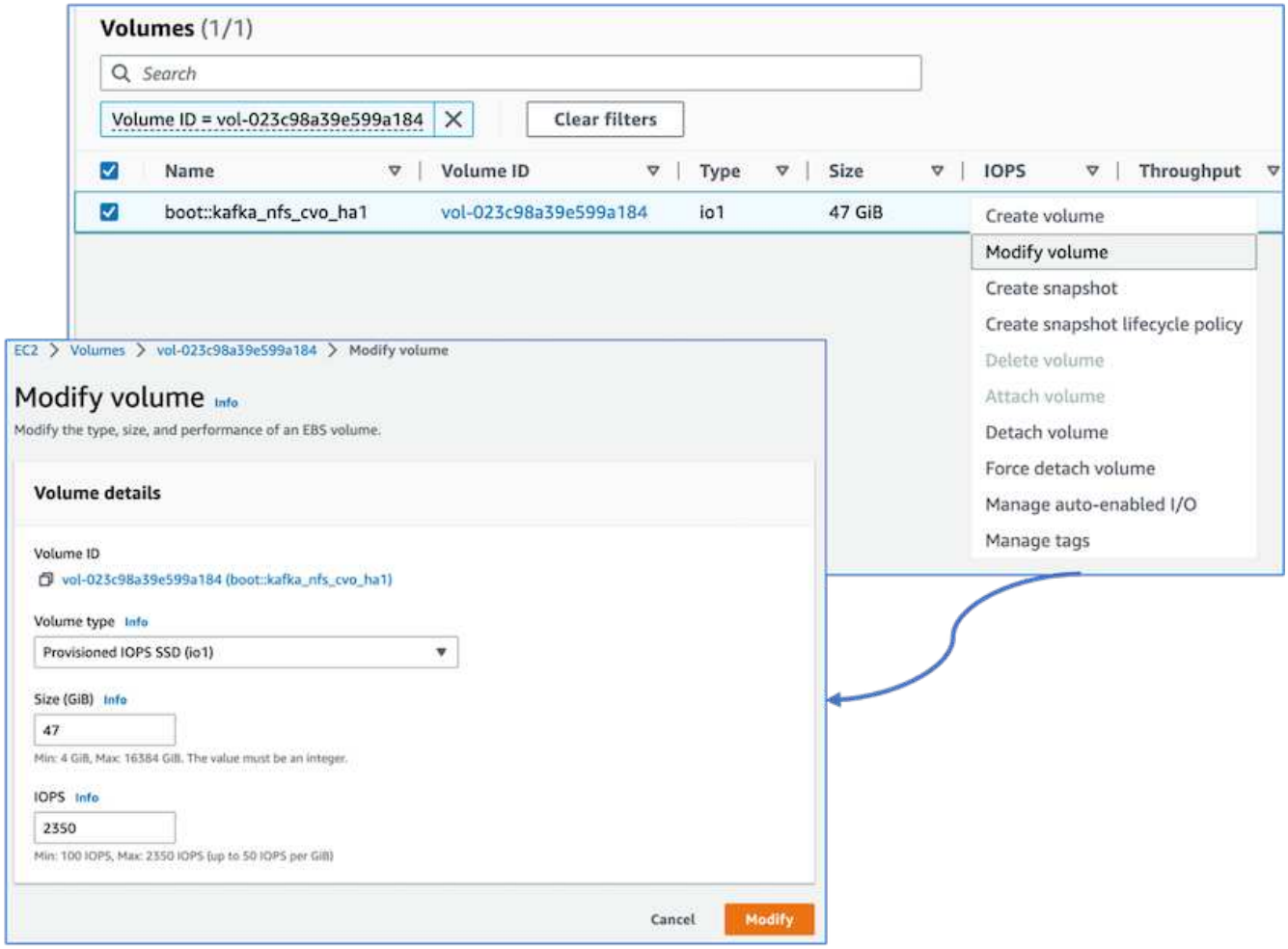


2. Para mejorar el rendimiento de red, hemos habilitado redes de alta velocidad para el par de alta disponibilidad y el nodo único.



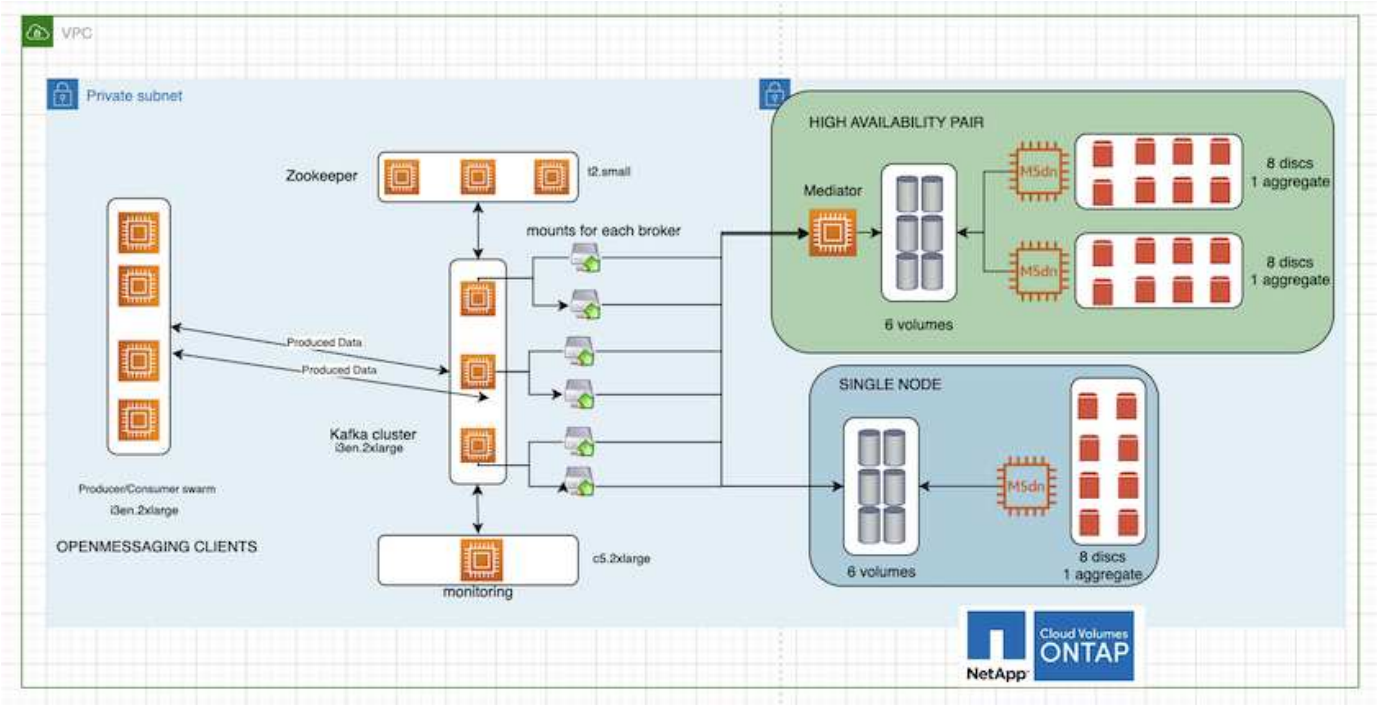
3. Hemos observado que el ONTAP NVRAM tenía más IOPS, por lo que hemos cambiado el IOPS a 2350 para el volumen raíz de Cloud Volumes ONTAP. El disco del volumen raíz en Cloud Volumes ONTAP tenía un tamaño de 47 GB. El siguiente comando ONTAP es para el par de alta disponibilidad, y el mismo paso es aplicable para el único nodo.

```
statistics start -object vnvram -instance vnvram -counter
backing_store_iops -sample-id sample_555
kafka_nfs_cvo_ha1::*> statistics show -sample-id sample_555
Object: vnvram
Instance: vnvram
Start-time: 1/18/2023 18:03:11
End-time: 1/18/2023 18:03:13
Elapsed-time: 2s
Scope: kafka_nfs_cvo_ha1-01
    Counter                                                     Value
    -------------------------------- --------------------------------
    backing_store_iops                                           1479
Object: vnvram
Instance: vnvram
Start-time: 1/18/2023 18:03:11
End-time: 1/18/2023 18:03:13
Elapsed-time: 2s
Scope: kafka_nfs_cvo_ha1-02
    Counter                                                     Value
    -------------------------------- --------------------------------
    backing_store_iops                                           1210
2 entries were displayed.
kafka_nfs_cvo_ha1::*>
```



En la figura siguiente se muestra la arquitectura de un clúster Kafka basado en NAS.

- **Compute.** utilizamos un clúster Kafka de tres nodos con un conjunto de zoomkeeper de tres nodos que se ejecuta en servidores dedicados. Cada agente tenía dos puntos de montaje NFS en un único volumen de la instancia de Cloud Volumes ONTAP a través de un LIF dedicado.
- **Supervisión.** utilizamos dos nodos para una combinación Prometheus-Grafana. Para generar cargas de trabajo, utilizamos un clúster de tres nodos independiente que podía producir y consumir este clúster Kafka.
- **Almacenamiento.** utilizamos una instancia Cloud Volumes ONTAP de par de alta disponibilidad con un volumen GP3 de 6 TB AWS-EBS montado en la instancia. El volumen se exportó después al agente de Kafka con un montaje NFS.



Configuraciones de pruebas de rendimiento de OpenMessage

1. Para obtener un mejor rendimiento NFS, se necesitan más conexiones de red entre el servidor NFS y el cliente NFS, que se pueden crear utilizando nconnect. Monte los volúmenes NFS en los nodos de broker con la opción nconnect ejecutando el siguiente comando:

```
[root@ip-172-30-0-121 ~]# cat /etc/fstab
UUID=eaa1f38e-de0f-4ed5-a5b5-2fa9db43bb38/xfsdefaults00
/dev/nvme1n1 /mnt/data-1 xfs defaults,noatime,nodiscard 0 0
/dev/nvme2n1 /mnt/data-2 xfs defaults,noatime,nodiscard 0 0
172.30.0.233:/kafka_aggr3_vol1 /kafka_aggr3_vol1 nfs
defaults,nconnect=16 0 0
172.30.0.233:/kafka_aggr3_vol2 /kafka_aggr3_vol2 nfs
defaults,nconnect=16 0 0
172.30.0.233:/kafka_aggr3_vol3 /kafka_aggr3_vol3 nfs
defaults,nconnect=16 0 0
172.30.0.242:/kafka_aggr22_vol1 /kafka_aggr22_vol1 nfs
defaults,nconnect=16 0 0
172.30.0.242:/kafka_aggr22_vol2 /kafka_aggr22_vol2 nfs
defaults,nconnect=16 0 0
172.30.0.242:/kafka_aggr22_vol3 /kafka_aggr22_vol3 nfs
defaults,nconnect=16 0 0
[root@ip-172-30-0-121 ~]# mount -a
[root@ip-172-30-0-121 ~]# df -h
Filesystem                       Size  Used Avail Use% Mounted on
devtmpfs                          31G     0   31G   0% /dev
tmpfs                             31G  249M   31G   1% /run
tmpfs                             31G     0   31G   0% /sys/fs/cgroup
/dev/nvme0n1p2                    10G  2.8G  7.2G  28% /
/dev/nvme1n1                     2.3T  248G  2.1T  11% /mnt/data-1
/dev/nvme2n1                     2.3T  245G  2.1T  11% /mnt/data-2
172.30.0.233:/kafka_aggr3_vol1   1.0T   12G 1013G   2% /kafka_aggr3_vol1
172.30.0.233:/kafka_aggr3_vol2   1.0T  5.5G 1019G   1% /kafka_aggr3_vol2
172.30.0.233:/kafka_aggr3_vol3   1.0T  8.9G 1016G   1% /kafka_aggr3_vol3
172.30.0.242:/kafka_aggr22_vol1  1.0T  7.3G 1017G   1%
/kafka_aggr22_vol1
172.30.0.242:/kafka_aggr22_vol2  1.0T  6.9G 1018G   1%
/kafka_aggr22_vol2
172.30.0.242:/kafka_aggr22_vol3  1.0T  5.9G 1019G   1%
/kafka_aggr22_vol3
tmpfs                            6.2G     0  6.2G   0% /run/user/1000
[root@ip-172-30-0-121 ~]#
```

2. Compruebe las conexiones de red en Cloud Volumes ONTAP. El siguiente comando ONTAP se usa desde el nodo único de Cloud Volumes ONTAP. El mismo paso es aplicable al par de alta disponibilidad de Cloud Volumes ONTAP.

```
Last login time: 1/20/2023 00:16:29
kafka_nfs_cvo_sn::> network connections active show -service nfs*
-fields remote-host
node                cid        vserver              remote-host
```

```
------------------- ---------- -------------------- ------------
kafka_nfs_cvo_sn-01 2315762628 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762629 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762630 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762631 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762632 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762633 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762634 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762635 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762636 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762637 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762639 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762640 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762641 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762642 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762643 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762644 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762645 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762646 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762647 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762648 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762649 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762650 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762651 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762652 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762653 svm_kafka_nfs_cvo_sn 172.30.0.121
kafka_nfs_cvo_sn-01 2315762656 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762657 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762658 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762659 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762660 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762661 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762662 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762663 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762664 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762665 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762666 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762667 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762668 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762669 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762670 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762671 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762672 svm_kafka_nfs_cvo_sn 172.30.0.72
kafka_nfs_cvo_sn-01 2315762673 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762674 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762676 svm_kafka_nfs_cvo_sn 172.30.0.121
```

```
kafka_nfs_cvo_sn-01 2315762677 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762678 svm_kafka_nfs_cvo_sn 172.30.0.223
kafka_nfs_cvo_sn-01 2315762679 svm_kafka_nfs_cvo_sn 172.30.0.223
48 entries were displayed.

kafka_nfs_cvo_sn::>
```

3. Utilizamos el siguiente Kafka `server.properties` En todos los agentes de Kafka para el par de alta disponibilidad de Cloud Volumes ONTAP. La `log.dirs` la propiedad es diferente para cada agente, y las propiedades restantes son comunes para los corredores. Para corredura1, el `log.dirs` el valor es el siguiente:

```
[root@ip-172-30-0-121 ~]# cat /opt/kafka/config/server.properties
broker.id=0
advertised.listeners=PLAINTEXT://172.30.0.121:9092
#log.dirs=/mnt/data-1/d1,/mnt/data-1/d2,/mnt/data-1/d3,/mnt/data-
2/d1,/mnt/data-2/d2,/mnt/data-2/d3
log.dirs=/kafka_aggr3_vol1/broker1,/kafka_aggr3_vol2/broker1,/kafka_aggr
3_vol3/broker1,/kafka_aggr22_vol1/broker1,/kafka_aggr22_vol2/broker1,/ka
fka_aggr22_vol3/broker1
zookeeper.connect=172.30.0.12:2181,172.30.0.30:2181,172.30.0.178:2181
num.network.threads=64
num.io.threads=64
socket.send.buffer.bytes=102400
socket.receive.buffer.bytes=102400
socket.request.max.bytes=104857600
num.partitions=1
num.recovery.threads.per.data.dir=1
offsets.topic.replication.factor=1
transaction.state.log.replication.factor=1
transaction.state.log.min.isr=1
replica.fetch.max.bytes=524288000
background.threads=20
num.replica.alter.log.dirs.threads=40
num.replica.fetchers=20
[root@ip-172-30-0-121 ~]#
```

   - Para corredura2, la `log.dirs` el valor de la propiedad es el siguiente:

```
log.dirs=/kafka_aggr3_vol1/broker2,/kafka_aggr3_vol2/broker2,/kafka_a
ggr3_vol3/broker2,/kafka_aggr22_vol1/broker2,/kafka_aggr22_vol2/broke
r2,/kafka_aggr22_vol3/broker2
```

   - Para corredura3, el `log.dirs` el valor de la propiedad es el siguiente:

```
log.dirs=/kafka_aggr3_vol1/broker3,/kafka_aggr3_vol2/broker3,/kafka_a
ggr3_vol3/broker3,/kafka_aggr22_vol1/broker3,/kafka_aggr22_vol2/broke
r3,/kafka_aggr22_vol3/broker3
```

4. Para el nodo único de Cloud Volumes ONTAP, el Kafka `servers.properties` Es lo mismo que para el par de alta disponibilidad de Cloud Volumes ONTAP, a excepción de la `log.dirs` propiedad.
   - Para corredura1, el `log.dirs` el valor es el siguiente:

     ```
     log.dirs=/kafka_aggr2_vol1/broker1,/kafka_aggr2_vol2/broker1,/kafka_a
     ggr2_vol3/broker1,/kafka_aggr2_vol4/broker1,/kafka_aggr2_vol5/broker1
     ,/kafka_aggr2_vol6/broker1
     ```

   - Para corredura2, la `log.dirs` el valor es el siguiente:

     ```
     log.dirs=/kafka_aggr2_vol1/broker2,/kafka_aggr2_vol2/broker2,/kafka_a
     ggr2_vol3/broker2,/kafka_aggr2_vol4/broker2,/kafka_aggr2_vol5/broker2
     ,/kafka_aggr2_vol6/broker2
     ```

   - Para corredura3, el `log.dirs` el valor de la propiedad es el siguiente:

     ```
     log.dirs=/kafka_aggr2_vol1/broker3,/kafka_aggr2_vol2/broker3,/kafka_a
     ggr2_vol3/broker3,/kafka_aggr2_vol4/broker3,/kafka_aggr2_vol5/broker3
     ,/kafka_aggr2_vol6/broker3
     ```

5. La carga de trabajo en el OMB se configura con las siguientes propiedades: `(/opt/benchmark/workloads/1-topic-100-partitions-1kb.yaml)`.

   ```
   topics: 4
   partitionsPerTopic: 100
   messageSize: 32768
   useRandomizedPayloads: true
   randomBytesRatio: 0.5
   randomizedPayloadPoolSize: 100
   subscriptionsPerTopic: 1
   consumerPerSubscription: 80
   producersPerTopic: 40
   producerRate: 1000000
   consumerBbacklogSizeGB: 0
   testDurationMinutes: 5
   ```

   La `messageSize` puede variar para cada caso de uso. En nuestra prueba de rendimiento, utilizamos 3K.

Utilizamos dos controladores distintos, Sync o Throughput, de OMB, para generar la carga de trabajo en el clúster Kafka.

- El archivo yaml utilizado para las propiedades del controlador Sync es el siguiente (`/opt/benchmark/driver- kafka/kafka-sync.yaml`):

```
name: Kafka
driverClass:
io.openmessaging.benchmark.driver.kafka.KafkaBenchmarkDriver
# Kafka client-specific configuration
replicationFactor: 3
topicConfig: |
  min.insync.replicas=2
  flush.messages=1
  flush.ms=0
commonConfig: |

bootstrap.servers=172.30.0.121:9092,172.30.0.72:9092,172.30.0.223:909
2
producerConfig: |
  acks=all
  linger.ms=1
  batch.size=1048576
consumerConfig: |
  auto.offset.reset=earliest
  enable.auto.commit=false
  max.partition.fetch.bytes=10485760
```

- El archivo yaml utilizado para las propiedades del controlador de rendimiento es el siguiente (`/opt/benchmark/driver- kafka/kafka-throughput.yaml`):

```
name: Kafka
driverClass:
io.openmessaging.benchmark.driver.kafka.KafkaBenchmarkDriver
# Kafka client-specific configuration
replicationFactor: 3
topicConfig: |
  min.insync.replicas=2
commonConfig: |

bootstrap.servers=172.30.0.121:9092,172.30.0.72:9092,172.30.0.223:909
2
  default.api.timeout.ms=1200000
  request.timeout.ms=1200000
producerConfig: |
  acks=all
  linger.ms=1
  batch.size=1048576
consumerConfig: |
  auto.offset.reset=earliest
  enable.auto.commit=false
  max.partition.fetch.bytes=10485760
```

Metodología de las pruebas

1. Se aprovisionó un clúster de Kafka según las especificaciones descritas anteriormente con Terraform y Ansible. Terraform se utiliza para crear la infraestructura con instancias de AWS para el clúster de Kafka y Ansible crea el clúster de Kafka.
2. Se activó una carga de trabajo de OMB con la configuración de carga de trabajo descrita anteriormente y el controlador de sincronización.

   ```
   Sudo bin/benchmark –drivers driver-kafka/kafka- sync.yaml workloads/1-
   topic-100-partitions-1kb.yaml
   ```

3. Se activó otra carga de trabajo con el controlador de rendimiento con la misma configuración de carga de trabajo.

   ```
   sudo bin/benchmark –drivers driver-kafka/kafka-throughput.yaml
   workloads/1-topic-100-partitions-1kb.yaml
   ```

Observación

Se utilizaron dos tipos distintos de controladores para generar cargas de trabajo con el fin de llevar a cabo una prueba de rendimiento de una instancia de Kafka que se ejecuta en NFS. La diferencia entre los controladores

es la propiedad log flush.

En un par de alta disponibilidad de Cloud Volumes ONTAP:

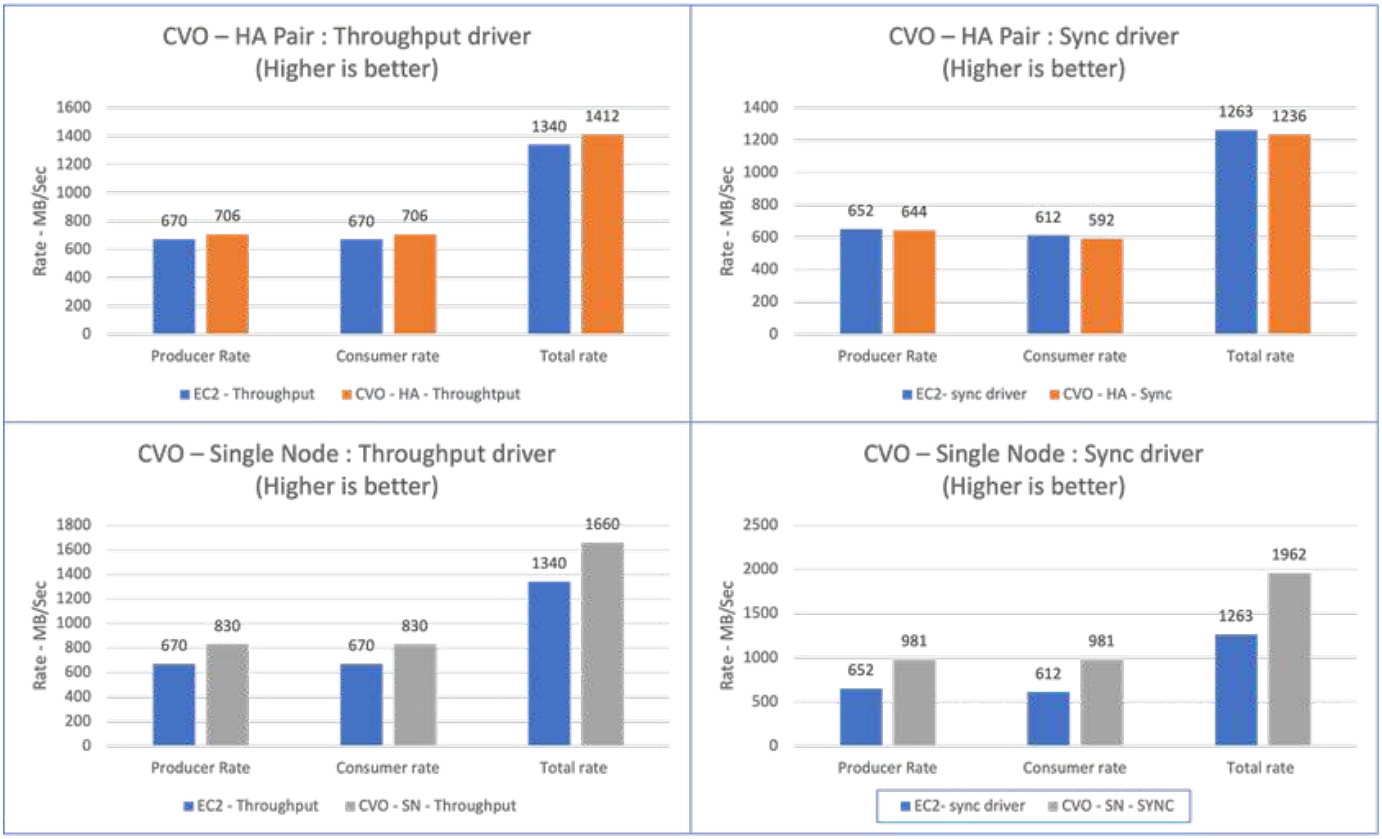
- Rendimiento total generado de forma coherente por el controlador Sync: ~1236 Mbps.
- Rendimiento total generado para el controlador de rendimiento: Pico de ~1412 Mbps.

Para un único nodo Cloud Volumes ONTAP:

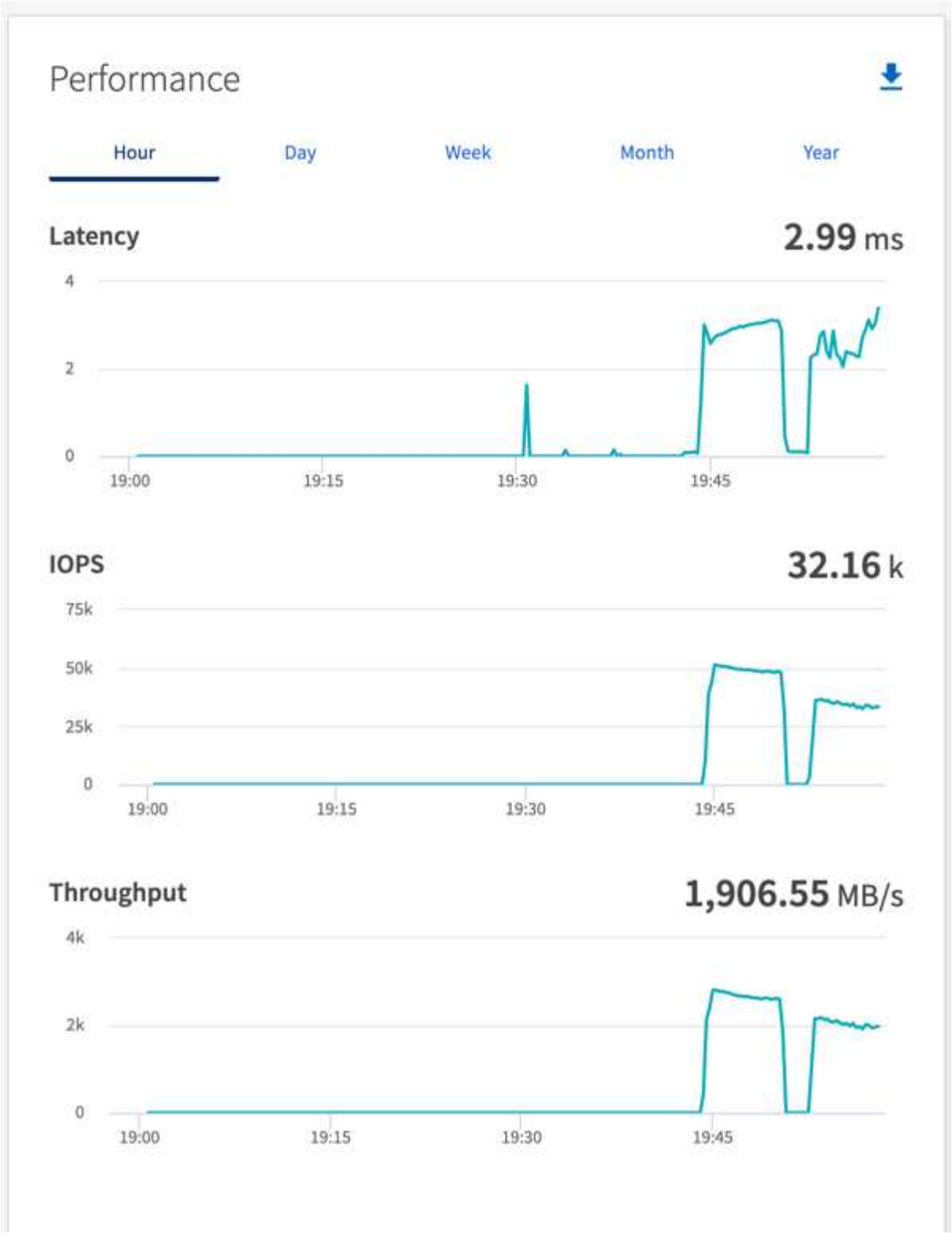
- Rendimiento total generado de forma consistente por el controlador Sync: ~ 1962MBps.
- Rendimiento total generado por el controlador de rendimiento: Pico de ~1660 MB

El controlador Sync puede generar un rendimiento constante a medida que los registros se vacíen en el disco al instante, mientras que el controlador de rendimiento genera ráfagas de rendimiento a medida que los registros se envían al disco de forma masiva.

Estos números de rendimiento se generan para la configuración de AWS determinada. Para requisitos de rendimiento más altos, los tipos de instancias se pueden escalar verticalmente para mejorar los números de rendimiento. El rendimiento total o la tasa total es la combinación de la tasa de producción y del consumidor.



Asegúrese de comprobar el rendimiento del almacenamiento al realizar el rendimiento o sincronizar las pruebas de rendimiento del controlador.



Información general sobre el rendimiento y la validación en AWS FSx para NetApp ONTAP

Se realizó una prueba de rendimiento de un clúster Kafka con la capa de almacenamiento montada en NFS de NetApp en AWS FSx para NetApp ONTAP. Los ejemplos de pruebas comparativas se describen en las siguientes secciones.

Apache Kafka en AWS FSx para NetApp ONTAP

Network File System (NFS) es un sistema de archivos de red ampliamente utilizado para almacenar grandes cantidades de datos. En la mayoría de las organizaciones, los datos se generan cada vez más a partir de aplicaciones de streaming como Apache Kafka. Estas cargas de trabajo requieren escalabilidad, baja latencia y una arquitectura de ingesta de datos sólida con funcionalidades de almacenamiento modernas. Para permitir el análisis en tiempo real y proporcionar información procesable, es necesaria una infraestructura bien diseñada y de alto rendimiento.

Kafka by design funciona con un sistema de archivos compatible con POSIX y se basa en el sistema de archivos para gestionar las operaciones de archivos, pero al almacenar datos en un sistema de archivos NFSv3, el cliente NFS de Kafka Broker puede interpretar las operaciones de archivos de forma diferente a un sistema de archivos local como XFS o Ext4. Un ejemplo común es el nombre de NFS Silly, que causó que los agentes de Kafka fallaran al expandir clústeres y volver a asignar particiones. Para hacer frente a este reto, NetApp ha actualizado el cliente NFS de Linux de código abierto con cambios que ahora están disponibles de forma general en RHEL8,7, RHEL9,1, y son compatibles con el lanzamiento actual de FSx para NetApp ONTAP, ONTAP 9.12.1.

Amazon FSx para NetApp ONTAP proporciona un sistema de archivos NFS en la nube totalmente gestionado, escalable y de alto rendimiento. Los datos de Kafka en FSx para NetApp ONTAP pueden escalarse para manejar grandes cantidades de datos y garantizar la tolerancia a fallos. NFS proporciona una gestión de almacenamiento centralizada y protección de datos para conjuntos de datos críticos y confidenciales.

Estas mejoras hacen posible que los clientes de AWS aprovechen FSx para NetApp ONTAP cuando ejecuten cargas de trabajo de Kafka en servicios de computación de AWS. Estos beneficios son:
* Reducir el uso de la CPU para reducir el tiempo de espera de E/S.
* Tiempo de recuperación de Kafka broker más rápido.
* Fiabilidad y eficiencia.
* Escalabilidad y rendimiento.
* Disponibilidad de la zona de multidisponibilidad.
* Protección de datos.

Información general sobre el rendimiento y la validación en AWS FSx para NetApp ONTAP

Se realizó una prueba de rendimiento de un clúster Kafka con capa de almacenamiento montada en NFS de NetApp en el cloud de AWS. Los ejemplos de pruebas comparativas se describen en las siguientes secciones.

Kafka en AWS FSx para NetApp ONTAP

Se realizó una prueba de rendimiento de un clúster de Kafka con AWS FSx para NetApp ONTAP en la nube de AWS. Esta evaluación comparativa se describe en las siguientes secciones.

Configuración de la arquitectura

La siguiente tabla muestra la configuración del entorno para un clúster Kafka usando AWS FSx para NetApp ONTAP.

Componente de plataforma	Configuración del entorno
Kafka 3.2.3	• 3 zookeepers – t2.pequeño • 3 servidores de broker: i3en.2xlarge • 1 x Grafana – c5n.2xgrande • 4 x productor/consumidor — c5n.2xgrande *
Sistema operativo en todos los nodos	RHEL8.6
AWS FSx para NetApp ONTAP	Multi-AZ con 4GB GB/s de rendimiento y 160000 000 IOPS

Configuración de NetApp FSx para NetApp ONTAP

1. Para nuestras pruebas iniciales, hemos creado un sistema de archivos FSx for NetApp ONTAP con 2TB TB de capacidad y 40000 000 IOPS para un rendimiento de 2GB GB/s.

   ```
   [root@ip-172-31-33-69 ~]# aws fsx create-file-system --region us-east-2
   --storage-capacity 2048 --subnet-ids <desired subnet 1> subnet-<desired
   subnet 2> --file-system-type ONTAP --ontap-configuration
   DeploymentType=MULTI_AZ_HA_1,ThroughputCapacity=2048,PreferredSubnetId=<
   desired primary subnet>,FsxAdminPassword=<new
   password>,DiskIopsConfiguration="{Mode=USER_PROVISIONED,Iops=40000"}
   ```

   En nuestro ejemplo, estamos poniendo en marcha FSx para NetApp ONTAP mediante la interfaz de línea de comandos de AWS. Tendrá que personalizar el comando de forma adicional en su entorno según sea necesario. FSX para NetApp ONTAP también se puede poner en marcha y gestionar a través de la consola de AWS para obtener una experiencia de puesta en marcha más sencilla y optimizada con menos entrada en la línea de comandos.

   Documentación En FSx para NetApp ONTAP, el número máximo de IOPS que se puede lograr para un sistema de archivos con un rendimiento de 2GB GB/s en nuestra región de prueba (EE. UU.-Este-1) es 80.000 iops. El número máximo total de iops para un sistema de archivos FSx para NetApp ONTAP es de 160.000 000 iops que requiere una puesta en marcha de 4GB GB/s de rendimiento para lograrlo, lo que demostraremos más adelante en este documento.

   Para obtener más información sobre las especificaciones de rendimiento de FSx para NetApp ONTAP, no dudes en visitar la documentación de AWS FSx para NetApp ONTAP aquí: https://docs.aws.amazon.com/fsx/latest/ONTAPGuide/performance.html .

   La sintaxis detallada de la línea de comandos para FSX “create-file-system” se puede encontrar aquí: https://docs.aws.amazon.com/cli/latest/reference/fsx/create-file-system.html

   Por ejemplo, puede especificar una clave KMS específica en lugar de la clave maestra predeterminada de AWS FSx que se utiliza cuando no se especifica ninguna clave KMS.

2. Al crear el sistema de archivos FSX for NetApp ONTAP, espere hasta que el estado de “ciclo de vida” cambie a “DISPONIBLE” en su retorno JSON después de describir su sistema de archivos de la siguiente manera:

```
[root@ip-172-31-33-69 ~]# aws fsx describe-file-systems  --region us-
east-1 --file-system-ids fs-02ff04bab5ce01c7c
```

3. Valide las credenciales iniciando sesión en FSx for NetApp ONTAP SSH con el usuario de fsxadmin: Fsxadmin es la cuenta de administrador predeterminada para FSX para sistemas de archivos NetApp ONTAP en la creación. La contraseña para fsxadmin es la contraseña que se configuró al crear por primera vez el sistema de archivos, ya sea en la consola de AWS o con la CLI de AWS, como se completó en el paso 1.

```
[root@ip-172-31-33-69 ~]# ssh fsxadmin@198.19.250.244
The authenticity of host '198.19.250.244 (198.19.250.244)' can't be
established.
ED25519 key fingerprint is
SHA256:mgCyRXJfWRc2d/jOjFbMBsUcYOWjxoIky0ltHvVDL/Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '198.19.250.244' (ED25519) to the list of
known hosts.
(fsxadmin@198.19.250.244) Password:

This is your first recorded login.
```

4. Una vez validadas sus credenciales, cree la máquina virtual de almacenamiento en el sistema de archivos FSx para NetApp ONTAP

```
[root@ip-172-31-33-69 ~]# aws fsx --region us-east-1 create-storage-
virtual-machine --name svmkafkatest --file-system-id fs-
02ff04bab5ce01c7c
```

Una máquina virtual de almacenamiento (SVM) es un servidor de archivos aislado con sus propias credenciales y extremos administrativos para administrar y acceder a los datos en FSx para volúmenes NetApp ONTAP y proporciona multi-tenancy de FSx para NetApp ONTAP.

5. Una vez que haya configurado su máquina virtual de almacenamiento principal, SSH en el sistema de archivos FSX for NetApp ONTAP recién creado y cree volúmenes en la máquina virtual de almacenamiento usando el comando de ejemplo siguiente y, de manera similar, creamos volúmenes 6 para esta validación. En base a nuestra validación, mantenga el componente predeterminado (8) o menos constituyentes que proporcionará un mejor rendimiento a kafka.

```
FsxId02ff04bab5ce01c7c::*> volume create -volume kafkafsxN1 -state
online -policy default -unix-permissions ---rwxr-xr-x -junction-active
true -type RW -snapshot-policy none  -junction-path /kafkafsxN1 -aggr
-list aggr1
```

6. Necesitaremos más capacidad en nuestros volúmenes para las pruebas. Amplíe el tamaño del volumen a 2TB TB y monte en la ruta de unión.

```
FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN1 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN1" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN2 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN2" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN3 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN3" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN4 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN4" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN5 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN5" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume size -volume kafkafsxN6 -new-size +2TB
vol size: Volume "svmkafkatest:kafkafsxN6" size set to 2.10t.

FsxId02ff04bab5ce01c7c::*> volume show -vserver svmkafkatest -volume *
Vserver   Volume       Aggregate    State      Type       Size
Available Used%
--------- ------------ ------------ ---------- ---- ----------
---------- -----
svmkafkatest
          kafkafsxN1   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          kafkafsxN2   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          kafkafsxN3   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          kafkafsxN4   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          kafkafsxN5   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          kafkafsxN6   -            online     RW       2.10TB
1.99TB    0%
svmkafkatest
          svmkafkatest_root
```

```
                        aggr1        online     RW          1GB
968.1MB    0%
7 entries were displayed.

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN1 -junction
-path /kafkafsxN1

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN2 -junction
-path /kafkafsxN2

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN3 -junction
-path /kafkafsxN3

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN4 -junction
-path /kafkafsxN4

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN5 -junction
-path /kafkafsxN5

FsxId02ff04bab5ce01c7c::*> volume mount -volume kafkafsxN6 -junction
-path /kafkafsxN6
```

En FSx para NetApp ONTAP, los volúmenes se pueden aprovisionar mediante thin provisioning. En nuestro ejemplo, la capacidad total de volumen extendido supera la capacidad total del sistema de archivos, por lo que necesitaremos ampliar la capacidad total del sistema de archivos para desbloquear la capacidad adicional de volumen aprovisionado que demostraremos en nuestro próximo paso.

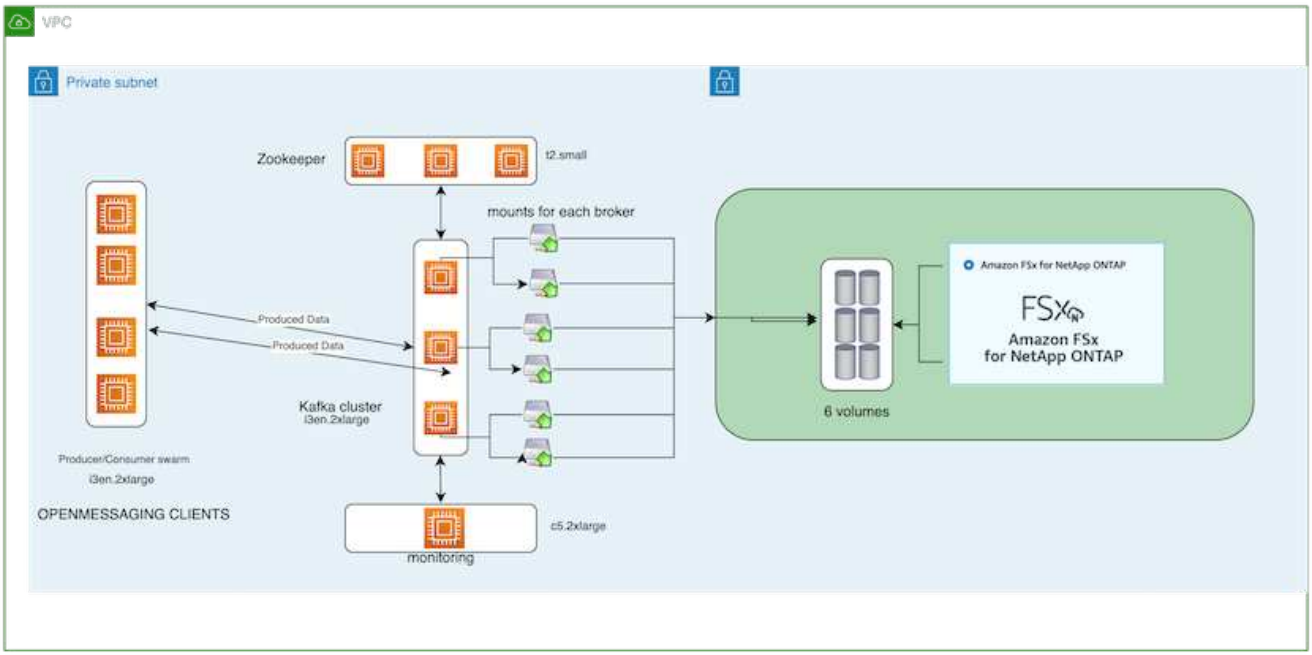
7. Después, para obtener más rendimiento y capacidad, ampliamos la capacidad de rendimiento de FSx para NetApp ONTAP de 2GB MB/s a 4GB MB/s y IOPS a 160000 GB, y la capacidad a 5 TB

```
[root@ip-172-31-33-69 ~]# aws fsx update-file-system --region us-east-1
--storage-capacity 5120 --ontap-configuration
'ThroughputCapacity=4096,DiskIopsConfiguration={Mode=USER_PROVISIONED,Io
ps=160000}' --file-system-id fs-02ff04bab5ce01c7c
```

La sintaxis detallada de la línea de comandos para FSX “update-file-system” se puede encontrar aquí: https://docs.aws.amazon.com/cli/latest/reference/fsx/update-file-system.html

8. El FSX para volúmenes NetApp ONTAP está montado con opiones predeterminados y nconnect en los brókeres kafka

En la siguiente imagen se muestra nuestra arquitectura final de un clúster kafka basado en FSx para NetApp ONTAP:



- Informática. Utilizamos un clúster Kafka de tres nodos con un conjunto de zookeeper de tres nodos ejecutándose en servidores dedicados. Cada agente tenía seis puntos de montaje de NFS en seis volúmenes en la instancia de FSx para NetApp ONTAP.
- Supervisión. Utilizamos dos nodos para una combinación de Prometheus-Grafana. Para generar cargas de trabajo, utilizamos un clúster de tres nodos independiente que podía producir y consumir este clúster Kafka.
- Reducida. Utilizamos un FSx para NetApp ONTAP con seis volúmenes de 2TB GB montados. A continuación, el volumen se exportó al agente Kafka con un montaje NFS. Los volúmenes FSx para NetApp ONTAP se montan con 16 sesiones nconnect y opciones predeterminadas en los agentes Kafka.

Configuraciones de OpenMessage Benchmarking.

Utilizamos la misma configuración utilizada para el NetApp Cloud Volumes ONTAP y sus detalles están aquí - https://docs.netapp.com/us-en/netapp-solutions/data-analytics/kafka-nfs-performance-overview-and-validation-in-aws.html#architectural-setup

Metodología de las pruebas

1. Un clúster de Kafka fue aprovisionado según la especificación descrita anteriormente usando terraform y ansible. Terraform se utiliza para construir la infraestructura utilizando instancias de AWS para el clúster Kafka y ansible construye el clúster Kafka en ellos.
2. Se activó una carga de trabajo de OMB con la configuración de carga de trabajo descrita anteriormente y el controlador de sincronización.

```
sudo bin/benchmark -drivers driver-kafka/kafka-sync.yaml workloads/1-
topic-100-partitions-1kb.yaml
```

3. Se activó otra carga de trabajo con el controlador de rendimiento con la misma configuración de carga de trabajo.

```
sudo bin/benchmark -drivers driver-kafka/kafka-throughput.yaml
workloads/1-topic-100-partitions-1kb.yaml
```

Observación

Se utilizaron dos tipos distintos de controladores para generar cargas de trabajo con el fin de llevar a cabo una prueba de rendimiento de una instancia de Kafka que se ejecuta en NFS. La diferencia entre los controladores es la propiedad log flush.

Para un factor de replicación Kafka 1 y FSx para NetApp ONTAP:

- Rendimiento total generado consistentemente por el controlador de sincronización: ~ 3218 Mbps y rendimiento máximo en ~ 3652 Mbps.
- Rendimiento total generado consistentemente por el controlador de rendimiento: ~ 3679 Mbps y rendimiento máximo en ~ 3908 Mbps.

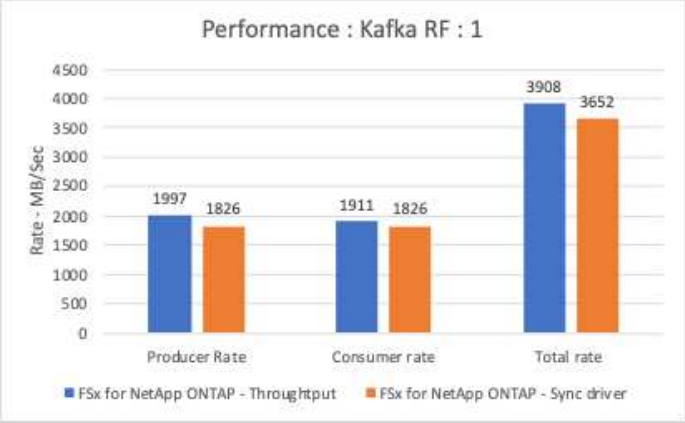
Para Kafka con factor de replicación 3 y FSx para NetApp ONTAP :

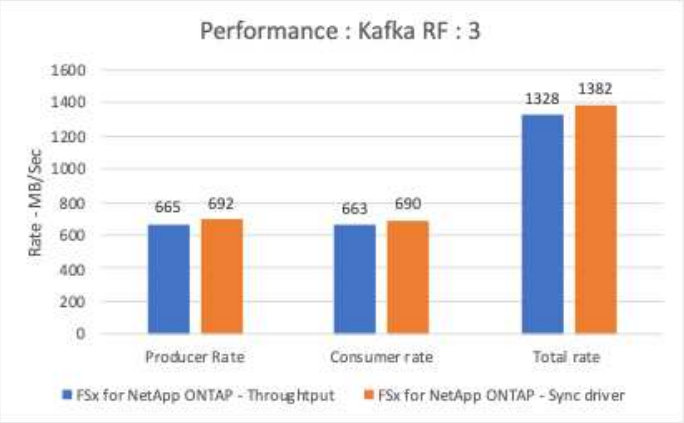
- Rendimiento total generado consistentemente por el controlador de sincronización: ~ 1252 Mbps y rendimiento máximo en ~ 1382 Mbps.
- Rendimiento total generado consistentemente por el controlador de rendimiento: ~ 1218 Mbps y rendimiento máximo en ~ 1328 Mbps.

En el factor de replicación Kafka 3, la operación de lectura y escritura se realizó tres veces en el FSX para NetApp ONTAP, en el factor de replicación Kafka 1, la operación de lectura y escritura es una vez en el FSX para NetApp ONTAP, por lo que en ambas validaciones, Somos capaces de alcanzar el rendimiento máximo de 4GB GB/s.

El controlador Sync puede generar un rendimiento constante a medida que los registros se vacíen en el disco al instante, mientras que el controlador de rendimiento genera ráfagas de rendimiento a medida que los registros se envían al disco de forma masiva.

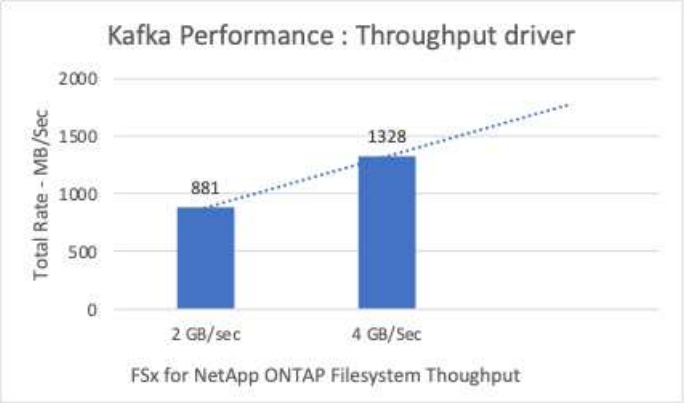
Estos números de rendimiento se generan para la configuración de AWS determinada. Para requisitos de rendimiento más altos, los tipos de instancias se pueden escalar verticalmente para mejorar los números de rendimiento. El rendimiento total o la tasa total es la combinación de la tasa de producción y del consumidor.

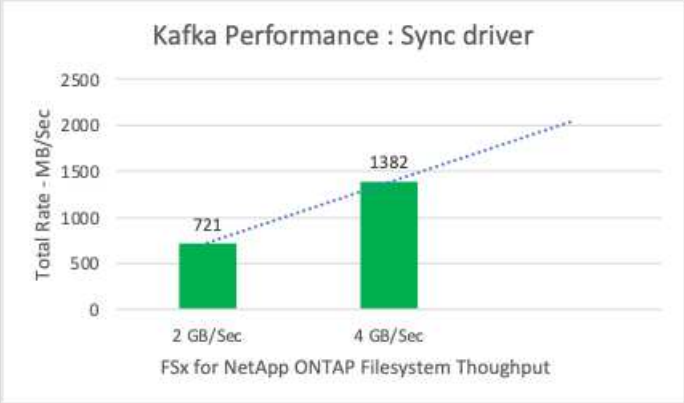




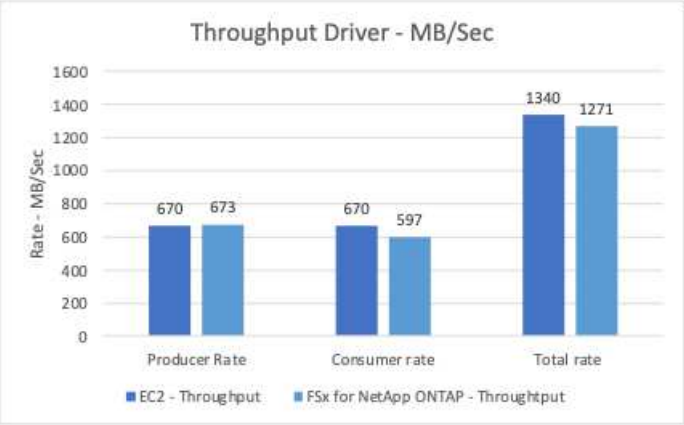
En el siguiente gráfico se muestra el rendimiento de 2GB MB/s FSx para NetApp ONTAP y 4GB GB/s para el factor de replicación Kafka 3. El factor de replicación 3 realiza la operación de lectura y escritura tres veces en

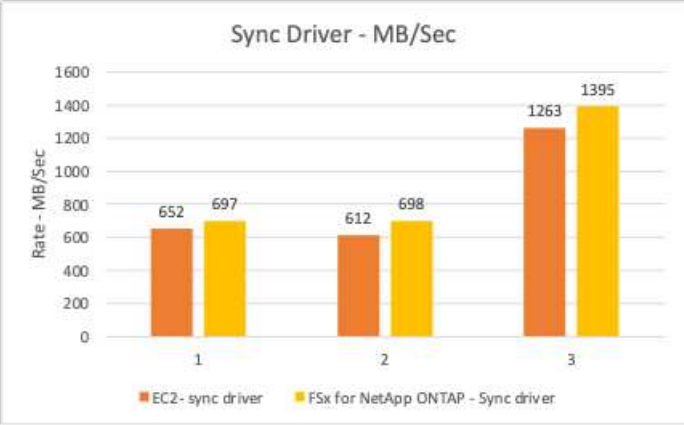
el almacenamiento FSx para NetApp ONTAP. La tasa total para el controlador de rendimiento es de 881 MB/s, que lee y escribe la operación Kafka aproximadamente 2,64 GB/s en el sistema de archivos FSX de 2GB MB/s para NetApp ONTAP y la tasa total para el controlador de rendimiento es de 1328 MB/s que lee y escribe la operación kafka aproximadamente 3,98 GB/s. El rendimiento de Kafka es lineal y escalable basado en el rendimiento de FSx para NetApp ONTAP.





En el siguiente gráfico se muestra el rendimiento entre la instancia de EC2 frente a FSx para NetApp ONTAP (Factor de replicación de Kafka: 3).

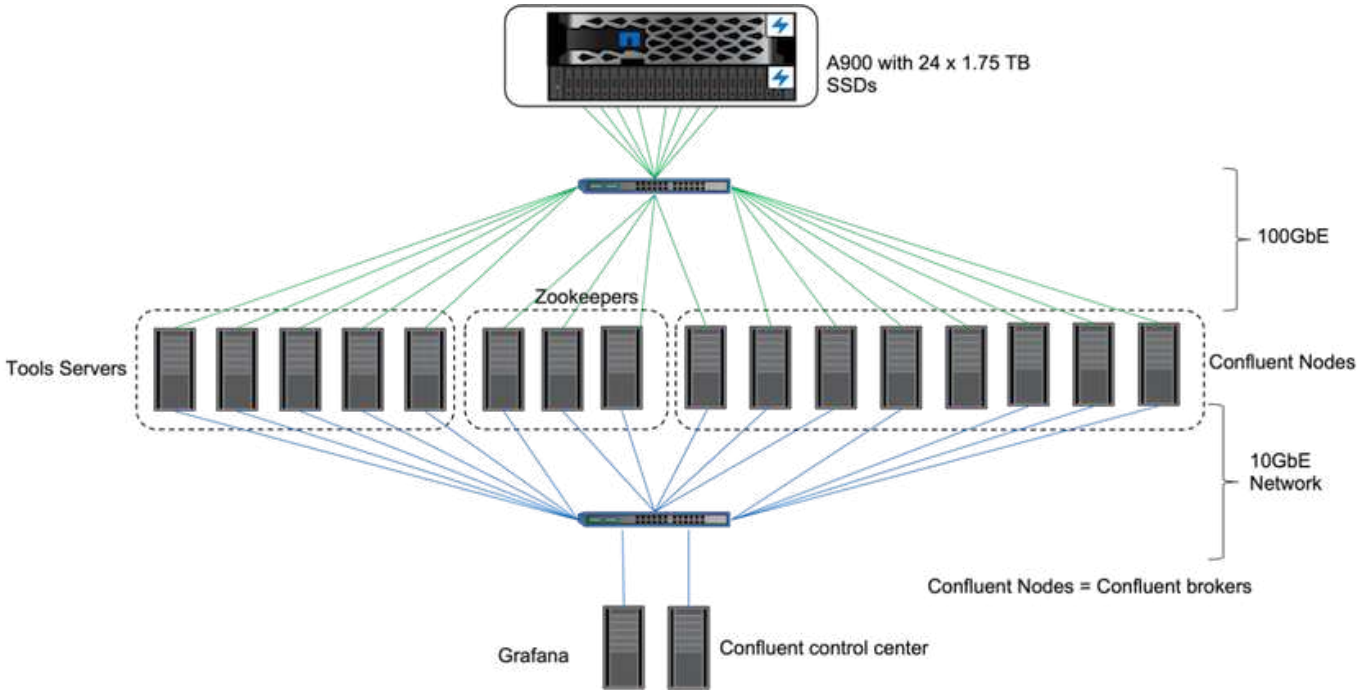




Descripción general y validación del rendimiento con AFF A900 en las instalaciones

En las instalaciones, utilizamos la controladora de almacenamiento AFF A900 de NetApp con ONTAP 9.12.1RC1 para validar el rendimiento y el escalado de un clúster Kafka. Utilizamos el mismo banco de pruebas que en nuestras prácticas recomendadas de almacenamiento por niveles anteriores con ONTAP y AFF.

Utilizamos Confluent Kafka 6.2.0 para evaluar el AFF A900. El clúster dispone de ocho nodos de broker y tres nodos de zomantenimiento. Para realizar pruebas de rendimiento, utilizamos cinco nodos de trabajo OMB.



Configuración del almacenamiento

Hemos utilizado instancias de NetApp FlexGroups para proporcionar un espacio de nombres único para directorios de registro, lo que simplifica la recuperación y la configuración. Hemos utilizado NFSv4.1 y pNFS para proporcionar acceso directo de ruta a los datos del segmento de registro.

Ajuste del cliente

Cada cliente montó la instancia de FlexGroup con el siguiente comando.

```
mount -t nfs -o vers=4.1,nconnect=16 172.30.0.121:/kafka_vol01
/data/kafka_vol01
```

Además, aumentamos la `max_session_slots` del valor predeterminado `64` para `180`. Esto coincide con el límite de ranuras de sesión predeterminado en ONTAP.

Ajuste de broker Kafka

Para maximizar el rendimiento en el sistema que se está probando, hemos aumentado de forma significativa los parámetros predeterminados para determinados grupos de subprocesos clave. Recomendamos seguir las prácticas recomendadas de Confluent Kafka para la mayoría de las configuraciones. Este ajuste se utilizó para maximizar la concurrencia de I/o excepcionales en el almacenamiento. Estos parámetros se pueden ajustar para ajustarse a los recursos informáticos y atributos de almacenamiento de su intermediario.

```
num.io.threads=96
num.network.threads=96
background.threads=20
num.replica.alter.log.dirs.threads=40
num.replica.fetchers=20
queued.max.requests=2000
```

Metodología de pruebas del generador de cargas de trabajo

Utilizamos las mismas configuraciones de OMB que para las pruebas en la nube para el controlador de rendimiento y la configuración de temas.

1. Se aprovisionó una instancia de FlexGroup con Ansible en un clúster de AFF.

```
---
- name: Set up kafka broker processes
  hosts: localhost
  vars:
    ntap_hostname: ‘hostname’
    ntap_username: 'user'
    ntap_password: 'password'
    size: 10
    size_unit: tb
    vserver: vs1
    state: present
    https: true
    export_policy: default
    volumes:
      - name: kafka_fg_vol01
        aggr: ["aggr1_a", "aggr2_a", “aggr1_b”, “aggr2_b”]
        path: /kafka_fg_vol01
  tasks:
    - name: Edit volumes
      netapp.ontap.na_ontap_volume:
        state: "{{ state }}"
        name: "{{ item.name }}"
        aggr_list: "{{ item.aggr }}"
        aggr_list_multiplier: 8
        size: "{{ size }}"
        size_unit: "{{ size_unit }}"
        vserver: "{{ vserver }}"
        snapshot_policy: none
        export_policy: default
        junction_path: "{{ item.path }}"
        qos_policy_group: none
        wait_for_completion: True
        hostname: "{{ ntap_hostname }}"
        username: "{{ ntap_username }}"
        password: "{{ ntap_password }}"
        https: "{{ https }}"
        validate_certs: false
      connection: local
      with_items: "{{ volumes }}"
```

2. PNFs se habilitó en la SVM de ONTAP.

```
vserver modify -vserver vs1 -v4.1-pnfs enabled -tcp-max-xfer-size 262144
```

3. La carga de trabajo se activó con el controlador de rendimiento utilizando la misma configuración de carga de trabajo que para Cloud Volumes ONTAP. Consulte la sección “Rendimiento en estado constante” más abajo. La carga de trabajo utilizó un factor de replicación de 3, lo que significa que se mantuvieron tres copias de segmentos de registro en NFS.

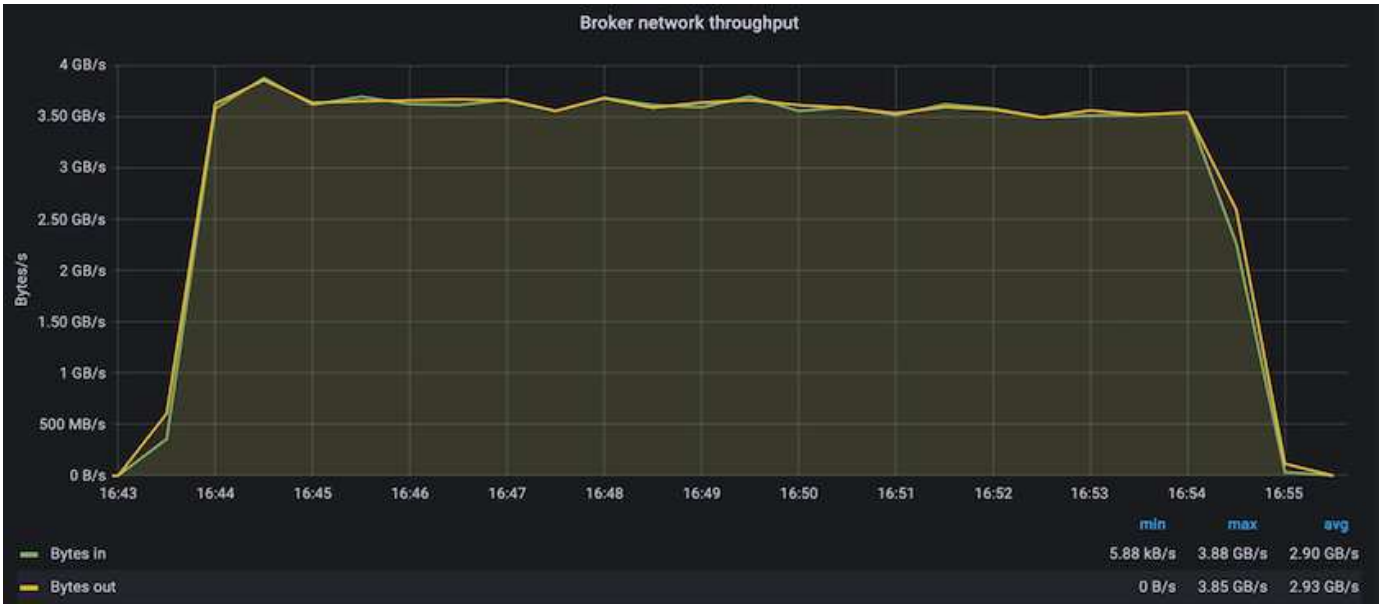
```
sudo bin/benchmark --drivers driver-kafka/kafka-throughput.yaml
workloads/1-topic-100-partitions-1kb.yaml
```

4. Por último, hemos completado las mediciones con un pedido atrasado para medir la capacidad de los consumidores para ponerse al día con los últimos mensajes. OMB construye un atraso al pausar a los consumidores durante el comienzo de una medición. Esto produce tres fases distintas: Creación de acumulación (tráfico sólo para productores), drenaje de acumulación (una fase de consumo pesado en la que los consumidores se quedan al tanto de los eventos perdidos en un tema) y el estado estable. Consulte la sección “análisis de los límites de almacenamiento” para más información.

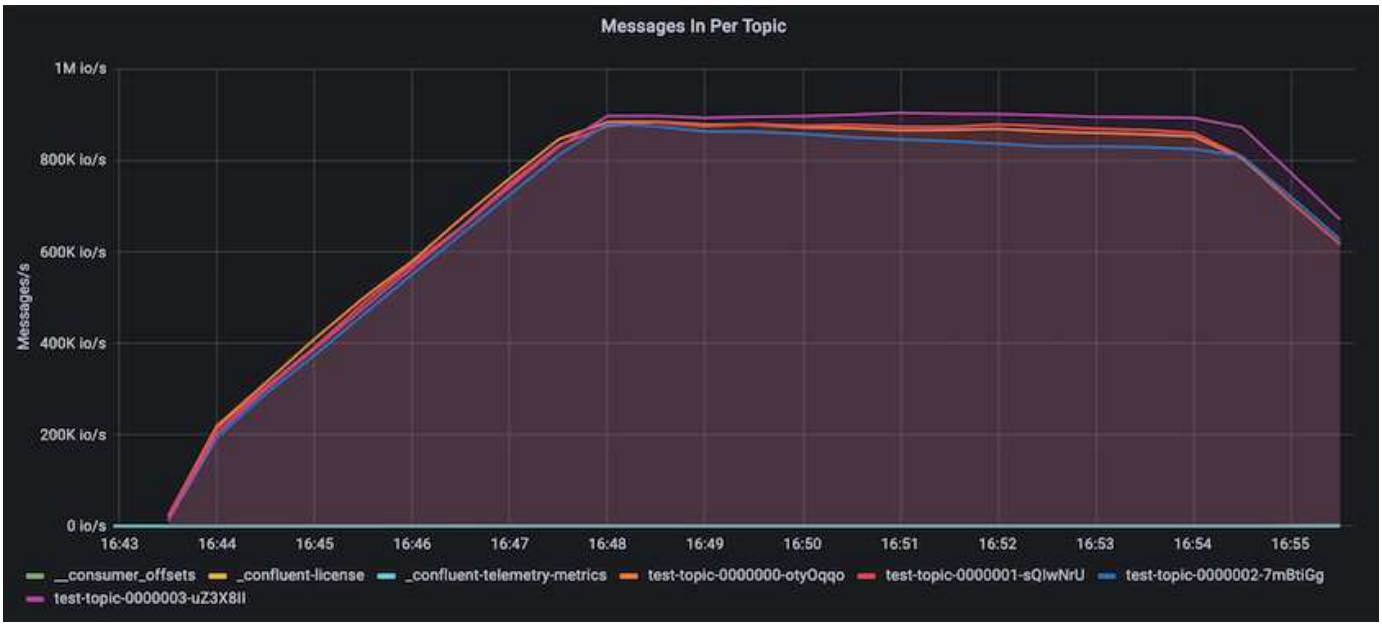
Rendimiento en estado constante

Evaluamos la plataforma AFF A900 utilizando la prueba de rendimiento de OpenMessaging para ofrecer una comparación similar a la de Cloud Volumes ONTAP en AWS y DAS en AWS. Todos los valores de rendimiento representan el rendimiento de Kafka-cluster a nivel de productor y consumidor.

El rendimiento constante con Confluent Kafka y AFF A900 logró un rendimiento medio de 3,4 Gbps tanto para el productor como para el consumidor. Esto es más de 3.4 millones de mensajes en el cluster Kafka. Al visualizar el rendimiento sostenido en bytes por segundo para BrokerTopicMetrics, observamos el excelente rendimiento en estado constante y el tráfico apoyado por la AFF A900.



Esto se alinea correctamente con la vista de los mensajes entregados por tema. El siguiente gráfico proporciona un desglose por tema. En la configuración probada, hemos visto unos 900 mensajes por tema en cuatro temas.

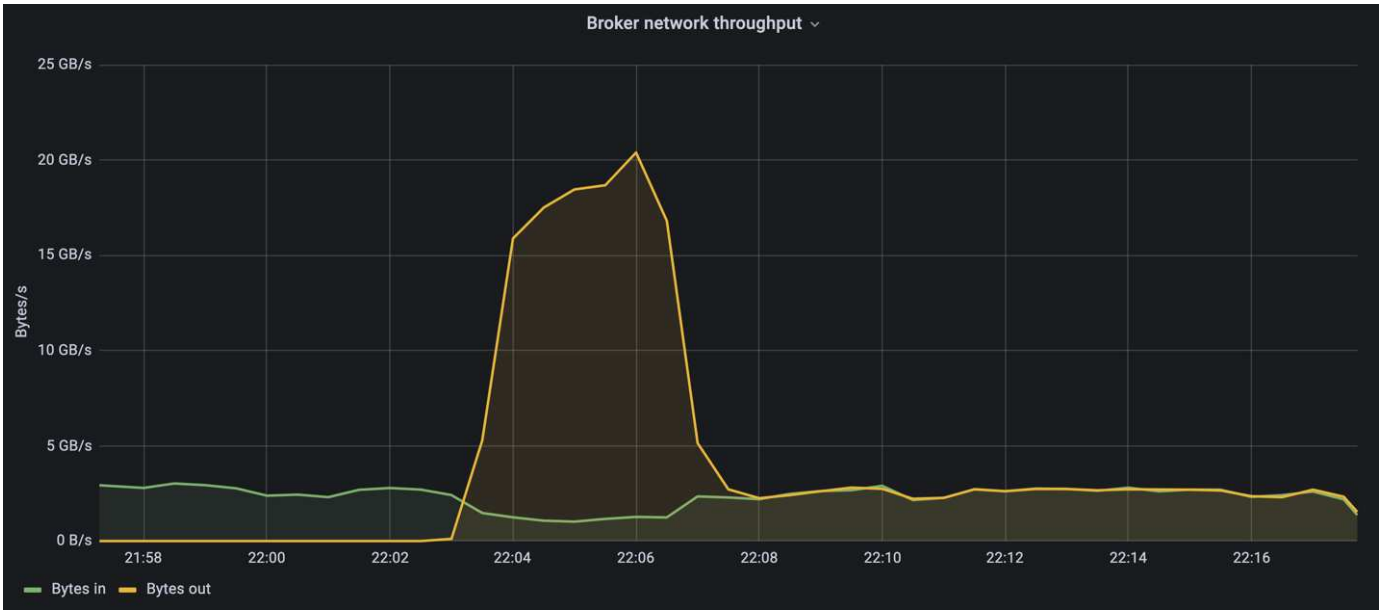


Rendimiento extremo y exploración de los límites de almacenamiento

Para AFF, también hemos probado con OMB mediante la función de acumulación. La función de acumulación detiene las suscripciones de consumidores mientras se crea una acumulación de eventos en el clúster Kafka. Durante esta fase, sólo se produce el tráfico de producción, que genera eventos que están comprometidos con los registros. De este modo, se emulan de forma más estrecha el procesamiento por lotes o los flujos de trabajo de análisis sin conexión; en estos flujos de trabajo, se inician las suscripciones de consumidores y se deben leer datos históricos que ya se han expulsado de la memoria caché de intermediarios.

Para comprender las limitaciones del almacenamiento en el rendimiento de consumo en esta configuración, medimos la fase de solo producción para comprender cuánto tráfico de escritura podría absorber A900. Consulte la siguiente sección “Orientación para la configuración” para comprender cómo aprovechar estos datos.

Durante la parte sólo para el productor de esta medición, observamos un alto rendimiento máximo que ha elevado los límites del rendimiento de A900 (cuando otros recursos de broker no estaban saturados al servicio del tráfico de productores y consumidores).



Hemos aumentado el tamaño del mensaje a 16 k para esta medición para limitar la sobrecarga por mensaje y maximizar la capacidad de almacenamiento a los puntos de montaje NFS.

```
messageSize: 16384
consumerBacklogSizeGB: 4096
```

El clúster Confluent Kafka logró un rendimiento máximo del productor de 4,03 Gbps.

```
18:12:23.833 [main] INFO WorkloadGenerator - Pub rate 257759.2 msg/s /
4027.5 MB/s | Pub err     0.0 err/s …
```

Una vez que OMB completó la acumulación de eventos, se reinició el tráfico de consumo. Durante las mediciones con drenaje de pedidos atrasados, observamos un rendimiento de consumo máximo de más de 20 Gbps en todos los temas. El rendimiento combinado que se aproximaba al volumen NFS donde se almacenaban los datos de registro de OMB era de unos 30 Gbps.

Orientación para la configuración

Amazon Web Services ofrece un "guía de tamaños" Para el ajuste de tamaño y el escalado de clústeres de Kafka.

Este ajuste de tamaño proporciona una fórmula útil para determinar los requisitos de rendimiento del almacenamiento para el clúster Kafka:

Para un rendimiento agregado producido en el clúster de tcluster con un factor de replicación de r, el rendimiento recibido por el almacenamiento de broker es el siguiente:

```
t[storage] = t[cluster]/#brokers + t[cluster]/#brokers * (r-1)
          = t[cluster]/#brokers * r
```

Esto se puede simplificar aún más:

```
max(t[cluster]) <= max(t[storage]) * #brokers/r
```

Con esta fórmula se puede seleccionar la plataforma ONTAP adecuada para las necesidades del nivel de sobrecarga Kafka.

En la siguiente tabla se explica el rendimiento previsto del productor para el A900 con diferentes factores de replicación:

Factor de replicación	Producción (GPPS)
3 (medidas)	3.4
2	5.1
1	10.2

Conclusión

La solución de NetApp para el tonto problema del cambio de nombre proporciona una forma de almacenamiento sencilla, económica y gestionada de forma centralizada para cargas de trabajo que antes eran incompatibles con NFS.

Este nuevo paradigma permite a los clientes crear clústeres Kafka más gestionables con más facilidad para migrar y hacer mirroring con fines específicos a efectos de la recuperación ante desastres y la protección de datos.
También hemos observado que NFS proporciona ventajas adicionales como un uso de CPU reducido y un tiempo de recuperación más rápido, una eficiencia del almacenamiento mejorada y un rendimiento mejorado con ONTAP de NetApp.

Dónde encontrar información adicional

Si quiere más información sobre el contenido de este documento, consulte los siguientes documentos o sitios web:

- ¿Qué es Apache Kafka?

  "https://www.confluent.io/what-is-apache-kafka/"

- ¿Qué es el tonto renombrar?

  "https://linux-nfs.org/wiki/index.php/Server-side_silly_rename"

- ONATP se lee para aplicaciones de transmisión.

  "https://www.netapp.com/blog/ontap-ready-for-streaming-applications/"

- Tonto- renombrar tema con Kafka.

  "https://sbg.technology/2018/07/10/kafka-nfs/"

- Documentación de productos de NetApp

  "https://www.netapp.com/support-and-training/documentation/"

- ¿Qué es NFS?

  "https://en.wikipedia.org/wiki/Network_File_System"

- ¿Qué es la reasignación de particiones Kafka?

  "https://docs.cloudera.com/runtime/7.2.10/kafka-managing/topics/kafka-manage-cli-reassign-overview.html"

- ¿Qué es la prueba de rendimiento de OpenMessaging?

  "https://openmessaging.cloud/"

- ¿Cómo se migra a un agente de Kafka?

  "https://medium.com/@sanchitbansal26/how-to-migrate-kafka-cluster-with-no-downtime-58c216129058"

- ¿Cómo se supervisa a Kafka Broker con Prometheus?

  https://www.confluent.io/blog/monitor-kafka-clusters-with-prometheus-grafana-and-confluent/

- Plataforma gestionada para Apache Kafka

  https://www.instaclustr.com/platform/managed-apache-kafka/

- Compatibilidad con Apache Kafka

  https://www.instaclustr.com/support-solutions/kafka-support/

- Servicios de consultoría de Apache Kafka

  https://www.instaclustr.com/services/consulting/

Información de copyright

Copyright © 2024 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en http://www.netapp.com/TM son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.