TR-4956: Automatisierte Bereitstellung von PostgreSQL mit hoher Verfügbarkeit und Notfallwiederherstellung in AWS FSx/EC2

NetApp database solutions

NetApp
August 18, 2025

Inhalt

TR-4956: Automatisierte Bereitstellung von PostgreSQL mit hoher Verfügbarkeit und Notfallwiederherstellung in AWS FSx/EC2

Allen Cao, Niyaz Mohamed, NetApp

Diese Lösung bietet einen Überblick und Details zur Bereitstellung von PostgreSQL-Datenbanken sowie zur Einrichtung von HA/DR, Failover und Resynchronisierung basierend auf der in das FSx ONTAP Speicherangebot integrierten NetApp SnapMirror -Technologie und dem NetApp Ansible-Automatisierungs-Toolkit in AWS.

Zweck

PostgreSQL ist eine weit verbreitete Open-Source-Datenbank, die auf Platz vier der zehn beliebtesten Datenbank-Engines steht."DB-Motoren" . Einerseits verdankt PostgreSQL seine Popularität seinem lizenzfreien Open-Source-Modell, das dennoch über anspruchsvolle Funktionen verfügt. Andererseits gibt es aufgrund der Open Source-Lösung keinen detaillierten Leitfaden zur produktionsreifen Datenbankbereitstellung im Bereich Hochverfügbarkeit und Notfallwiederherstellung (HA/DR), insbesondere in der öffentlichen Cloud. Im Allgemeinen kann es schwierig sein, ein typisches PostgreSQL HA/DR-System mit Hot- und Warm-Standby, Streaming-Replikation usw. einzurichten. Das Testen der HA/DR-Umgebung durch Hochstufen der Standby-Site und anschließendes Zurückschalten auf die primäre Site kann die Produktion stören. Es gibt gut dokumentierte Leistungsprobleme auf dem Primärrechner, wenn Lese-Workloads auf Streaming-Hot-Standby bereitgestellt werden.

In dieser Dokumentation zeigen wir, wie Sie auf eine PostgreSQL-Streaming-HA/DR-Lösung auf Anwendungsebene verzichten und mithilfe der Replikation auf Speicherebene eine PostgreSQL-HA/DR-Lösung basierend auf AWS FSx ONTAP -Speicher und EC2-Recheninstanzen erstellen können. Die Lösung erstellt ein einfacheres und vergleichbares System und liefert im Vergleich zur herkömmlichen PostgreSQL-Streaming-Replikation auf Anwendungsebene für HA/DR gleichwertige Ergebnisse.

Diese Lösung basiert auf der bewährten und ausgereiften NetApp SnapMirror -Replikationstechnologie auf Speicherebene, die im AWS-nativen FSX ONTAP Cloud-Speicher für PostgreSQL HA/DR verfügbar ist. Die Implementierung ist mit einem Automatisierungs-Toolkit des NetApp Solutions-Teams ganz einfach. Es bietet ähnliche Funktionen und beseitigt gleichzeitig die Komplexität und Leistungseinbußen auf der primären Site mit der auf Streaming basierenden HA/DR-Lösung auf Anwendungsebene. Die Lösung kann einfach bereitgestellt und getestet werden, ohne den aktiven primären Standort zu beeinträchtigen.

Diese Lösung ist für die folgenden Anwendungsfälle geeignet:

- Produktionsreife HA/DR-Bereitstellung für PostgreSQL in der öffentlichen AWS-Cloud
- Testen und Validieren einer PostgreSQL-Workload in der öffentlichen AWS-Cloud
- Testen und Validieren einer PostgreSQL HA/DR-Strategie basierend auf der NetApp SnapMirror Replikationstechnologie

Publikum

Diese Lösung ist für folgende Personen gedacht:

- Der DBA, der an der Bereitstellung von PostgreSQL mit HA/DR in der öffentlichen AWS-Cloud interessiert

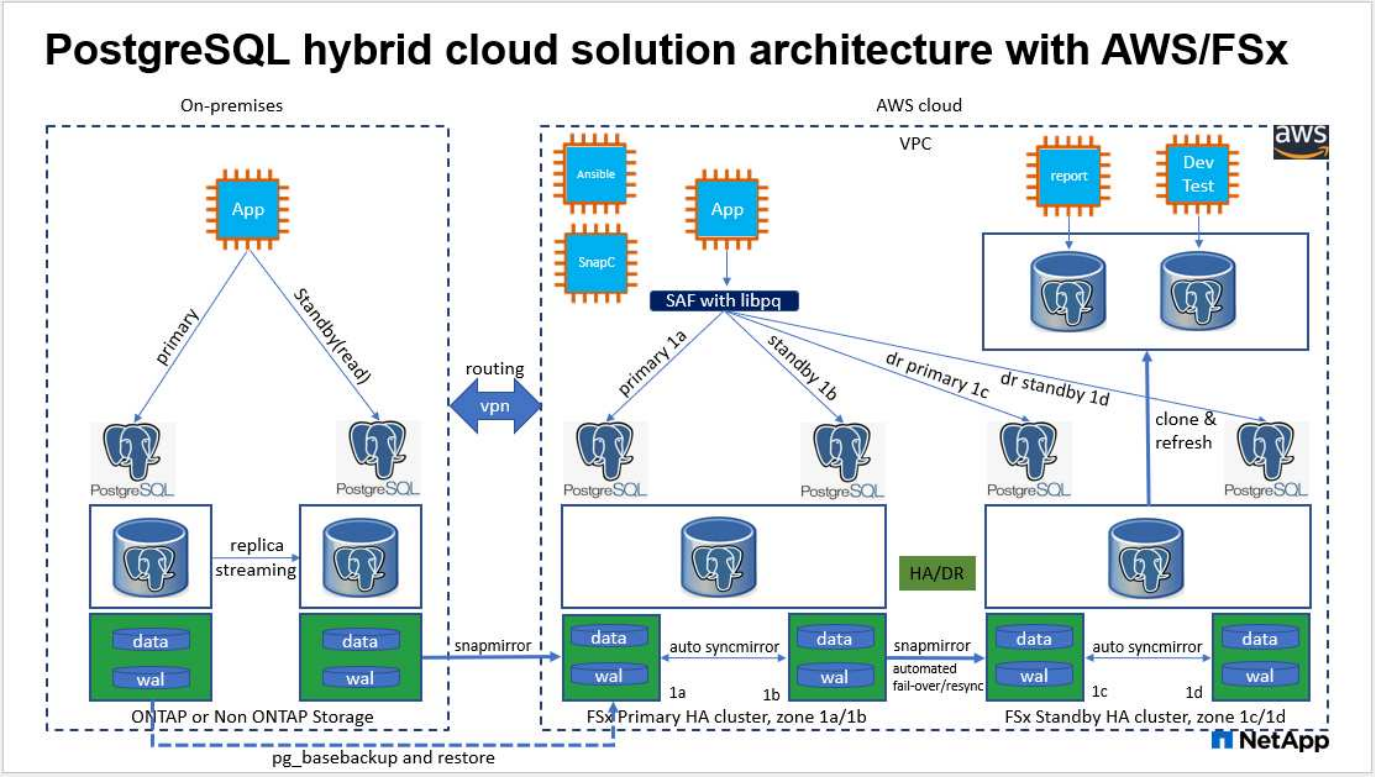
ist.

- Der Datenbanklösungsarchitekt, der daran interessiert ist, PostgreSQL-Workloads in der öffentlichen AWS-Cloud zu testen.
- Der Speicheradministrator, der an der Bereitstellung und Verwaltung von PostgreSQL-Instanzen interessiert ist, die im AWS FSx-Speicher bereitgestellt werden.
- Der Anwendungsbesitzer, der daran interessiert ist, eine PostgreSQL-Umgebung in AWS FSx/EC2 einzurichten.

Test- und Validierungsumgebung für Lösungen

Das Testen und Validieren dieser Lösung wurde in einer AWS FSx- und EC2-Umgebung durchgeführt, die möglicherweise nicht der endgültigen Bereitstellungsumgebung entspricht. Weitere Informationen finden Sie im Abschnitt Wichtige Faktoren für die Bereitstellungsüberlegungen .

Architektur



Hardware- und Softwarekomponenten

Hardware		
FSx ONTAP Speicher	Aktuelle Version	Zwei FSx HA-Paare in derselben VPC und Verfügbarkeitszone als primäre und Standby-HA-Cluster
EC2-Instanz für Compute	t2.xlarge/4vCPU/16G	Zwei EC2 T2 xlarge als primäre und Standby-Compute-Instanzen

Ansible-Controller	Centos VM/4vCPU/8G vor Ort	Eine VM zum Hosten des Ansible-Automatisierungscontrollers entweder vor Ort oder in der Cloud
Software		
RedHat Linux	RHEL-8.6.0_HVM-20220503-x86_64-2-Hourly2-GP2	RedHat-Abonnement zum Testen bereitgestellt
Centos Linux	CentOS Linux Version 8.2.2004 (Core)	Hosten eines im lokalen Labor bereitgestellten Ansible-Controllers
PostgreSQL	Version 14.5	Die Automatisierung ruft die neueste verfügbare Version von PostgreSQL aus dem postgresql.ora-Yum-Repository ab.
Ansible	Version 2.10.3	Voraussetzungen für erforderliche Sammlungen und Bibliotheken, die mit dem Anforderungs-Playbook installiert wurden

Wichtige Faktoren für die Bereitstellungsüberlegungen

- **Sicherung, Wiederherstellung und Wiederherstellung der PostgreSQL-Datenbank.** Eine PostgreSQL-Datenbank unterstützt eine Reihe von Sicherungsmethoden, z. B. eine logische Sicherung mit pg_dump, eine physische Online-Sicherung mit pg_basebackup oder einem Betriebssystem-Sicherungsbefehl auf niedrigerer Ebene sowie speicherebenenkonsistente Snapshots. Diese Lösung verwendet Snapshots von NetApp -Konsistenzgruppen für die Sicherung, Wiederherstellung und Wiederherstellung von PostgreSQL-Datenbankdaten und WAL-Volumes am Standby-Standort. Die Volume-Snapshots der NetApp -Konsistenzgruppe sequenzieren die E/A-Vorgänge beim Schreiben in den Speicher und schützen die Integrität der Datenbankdatendateien.
- **EC2-Recheninstanzen.** Bei diesen Tests und Validierungen haben wir den AWS EC2 t2.xlarge-Instanztyp für die PostgreSQL-Datenbank-Compute-Instanz verwendet. NetApp empfiehlt die Verwendung einer EC2-Instanz vom Typ M5 als Compute-Instanz für PostgreSQL bei der Bereitstellung, da diese für Datenbank-Workloads optimiert ist. Die Standby-Compute-Instanz sollte immer in derselben Zone bereitgestellt werden wie das passive (Standby-)Dateisystem, das für den FSx HA-Cluster bereitgestellt wird.
- **FSx-Speicher-HA-Cluster, Bereitstellung in einer oder mehreren Zonen.** Bei diesen Tests und Validierungen haben wir einen FSx HA-Cluster in einer einzelnen AWS-Verfügbarkeitszone bereitgestellt. Für die Produktionsbereitstellung empfiehlt NetApp die Bereitstellung eines FSx HA-Paares in zwei verschiedenen Verfügbarkeitszonen. Ein Disaster-Recovery-Standby-HA-Paar für die Geschäftskontinuität kann in einer anderen Region eingerichtet werden, wenn zwischen dem Primär- und dem Standby-System ein bestimmter Abstand erforderlich ist. Ein FSx HA-Cluster wird immer in einem HA-Paar bereitgestellt, das in einem Paar aktiv-passiver Dateisysteme synchron gespiegelt wird, um Redundanz auf Speicherebene bereitzustellen.
- **PostgreSQL-Daten- und Protokollplatzierung.** Typische PostgreSQL-Bereitstellungen nutzen dasselbe Stammverzeichnis oder dieselben Volumes für Daten- und Protokolldateien. In unseren Tests und Validierungen haben wir PostgreSQL-Daten und -Protokolle aus Leistungsgründen in zwei separate Volumes aufgeteilt. Im Datenverzeichnis wird ein Softlink verwendet, um auf das Protokollverzeichnis oder Volume zu verweisen, das PostgreSQL-WAL-Protokolle und archivierte WAL-Protokolle hostet.
- **Zeitgeber für die Verzögerung beim Start des PostgreSQL-Dienstes.** Diese Lösung verwendet NFS-gemountete Volumes zum Speichern der PostgreSQL-Datenbankdatei und der WAL-Protokolldateien. Während eines Neustarts des Datenbankhosts versucht der PostgreSQL-Dienst möglicherweise zu starten, während das Volume nicht gemountet ist. Dies führt zu einem Startfehler des Datenbankdienstes.

Damit die PostgreSQL-Datenbank korrekt gestartet werden kann, ist eine Zeitverzögerung von 10 bis 15 Sekunden erforderlich.

- **RPO/RTO für Geschäftskontinuität.** Die FSx-Datenreplikation vom Primär- zum Standby-System für DR basiert auf ASYNC, was bedeutet, dass das RPO von der Häufigkeit der Snapshot-Backups und der SnapMirror Replikation abhängt. Eine höhere Häufigkeit von Snapshot-Kopien und SnapMirror -Replikation reduziert das RPO. Daher besteht ein Gleichgewicht zwischen dem potenziellen Datenverlust im Katastrophenfall und den zusätzlichen Speicherkosten. Wir haben festgestellt, dass Snapshot-Kopien und SnapMirror Replikationen für RPO in Intervallen von nur 5 Minuten implementiert werden können und PostgreSQL für RTO im Allgemeinen in weniger als einer Minute am DR-Standby-Standort wiederhergestellt werden kann.
- **Datenbanksicherung.** Nachdem eine PostgreSQL-Datenbank implementiert oder von einem lokalen Rechenzentrum in den AWS FSx-Speicher migriert wurde, werden die Daten zum Schutz automatisch synchronisiert und im FSx HA-Paar gespiegelt. Im Katastrophenfall werden die Daten zusätzlich durch einen replizierten Standby-Standort geschützt. Für eine längerfristige Backup-Aufbewahrung oder Datensicherung empfiehlt NetApp die Verwendung des integrierten PostgreSQL-Dienstprogramms pg_basebackup, um ein vollständiges Datenbank-Backup auszuführen, das auf den S3-Blob-Speicher portiert werden kann.

Lösungsbereitstellung

Die Bereitstellung dieser Lösung kann mithilfe des auf NetApp Ansible basierenden Automatisierungs-Toolkits automatisch abgeschlossen werden, indem Sie die unten aufgeführten detaillierten Anweisungen befolgen.

1. Lesen Sie die Anweisungen im Automatisierungs-Toolkit READme.md"na_postgresql_aws_deploy_hadr" .
2. Sehen Sie sich das folgende Video an.

Automatisierte Bereitstellung und Schutz von PostgreSQL

1. Konfigurieren Sie die erforderlichen Parameterdateien(`hosts` , `host_vars/host_name.yml` , `fsx_vars.yml`), indem Sie in den entsprechenden Abschnitten der Vorlage benutzerspezifische Parameter eingeben. Verwenden Sie dann die Schaltfläche „Kopieren“, um Dateien auf den Ansible-Controller-Host zu kopieren.

Voraussetzungen für die automatisierte Bereitstellung

Für die Bereitstellung sind die folgenden Voraussetzungen erforderlich.

1. Ein AWS-Konto wurde eingerichtet und die erforderlichen VPC- und Netzwerksegmente wurden innerhalb Ihres AWS-Kontos erstellt.
2. Von der AWS EC2-Konsole aus müssen Sie zwei EC2-Linux-Instanzen bereitstellen, eine als primären PostgreSQL-DB-Server am primären und eine am Standby-DR-Standort. Stellen Sie zur Rechenredundanz an den primären und Standby-DR-Standorten zwei zusätzliche EC2-Linux-Instanzen als Standby-PostgreSQL-DB-Server bereit. Weitere Einzelheiten zur Umgebungseinrichtung finden Sie im Architekturdiagramm im vorherigen Abschnitt. Überprüfen Sie auch die"Benutzerhandbuch für Linux-Instanzen" für weitere Informationen.
3. Stellen Sie über die AWS EC2-Konsole zwei FSx ONTAP Speicher-HA-Cluster bereit, um die PostgreSQL-Datenbankvolumes zu hosten. Wenn Sie mit der Bereitstellung von FSx-Speicher nicht vertraut sind, lesen Sie die Dokumentation"Erstellen von FSx ONTAP Dateisystemen" für schrittweise Anleitungen.
4. Erstellen Sie eine Centos Linux-VM zum Hosten des Ansible-Controllers. Der Ansible-Controller kann sich entweder vor Ort oder in der AWS-Cloud befinden. Wenn es sich vor Ort befindet, müssen Sie über eine SSH-Verbindung zur VPC, zu EC2-Linux-Instanzen und zu FSx-Speicherclustern verfügen.

5. Richten Sie den Ansible-Controller wie im Abschnitt „Einrichten des Ansible-Steuerknotens für CLI-Bereitstellungen auf RHEL/CentOS“ aus der Ressource ein."Erste Schritte mit der NetApp Lösungsautomatisierung" .
6. Klonen Sie eine Kopie des Automatisierungs-Toolkits von der öffentlichen NetApp GitHub-Site.

```
git clone https://github.com/NetApp-
Automation/na_postgresql_aws_deploy_hadr.git
```

1. Führen Sie aus dem Stammverzeichnis des Toolkits die erforderlichen Playbooks aus, um die erforderlichen Sammlungen und Bibliotheken für den Ansible-Controller zu installieren.

```
ansible-playbook -i hosts requirements.yml
```

```
ansible-galaxy collection install -r collections/requirements.yml --force
--force-with-deps
```

1. Rufen Sie die erforderlichen EC2 FSx-Instanzparameter für die DB-Hostvariablendatei ab `host_vars/*` und die globale Variablendatei `fsx_vars.yml` Konfiguration.

Konfigurieren der Hosts-Datei

Geben Sie die primäre FSx ONTAP Clusterverwaltungs-IP und die Hostnamen der EC2-Instanzen in die Hosts-Datei ein.

```
# Primary FSx cluster management IP address
[fsx_ontap]
172.30.15.33
```

```
# Primary PostgreSQL DB server at primary site where database is
initialized at deployment time
[postgresql]
psql_01p ansible_ssh_private_key_file=psql_01p.pem
```

```
# Primary PostgreSQL DB server at standby site where postgresql service is
installed but disabled at deployment
# Standby DB server at primary site, to setup this server comment out
other servers in [dr_postgresql]
# Standby DB server at standby site, to setup this server comment out
other servers in [dr_postgresql]
[dr_postgresql] --
psql_01s ansible_ssh_private_key_file=psql_01s.pem
#psql_01ps ansible_ssh_private_key_file=psql_01ps.pem
#psql_01ss ansible_ssh_private_key_file=psql_01ss.pem
```

Konfigurieren Sie die Datei host_name.yml im Ordner host_vars

```
# Add your AWS EC2 instance IP address for the respective PostgreSQL
server host
ansible_host: "10.61.180.15"

# "{{groups.postgresql[0]}}" represents first PostgreSQL DB server as
defined in PostgreSQL hosts group [postgresql]. For concurrent multiple
PostgreSQL DB servers deployment, [0] will be incremented for each
additional DB server. For example,  "{{groups.posgresql[1]}}" represents
DB server 2, "{{groups.posgresql[2]}}" represents DB server 3 ... As a
good practice and the default, two volumes are allocated to a PostgreSQL
DB server with corresponding /pgdata, /pglogs mount points, which store
PostgreSQL data, and PostgreSQL log files respectively. The number and
naming of DB volumes allocated to a DB server must match with what is
defined in global fsx_vars.yml file by src_db_vols, src_archivelog_vols
parameters, which dictates how many volumes are to be created for each DB
server. aggr_name is aggr1 by default. Do not change. lif address is the
NFS IP address for the SVM where PostgreSQL server is expected to mount
its database volumes. Primary site servers from primary SVM and standby
servers from standby SVM.
host_datastores_nfs:
  - {vol_name: "{{groups.postgresql[0]}}_pgdata", aggr_name: "aggr1", lif:
"172.21.94.200", size: "100"}
  - {vol_name: "{{groups.postgresql[0]}}_pglogs", aggr_name: "aggr1", lif:
"172.21.94.200", size: "100"}

# Add swap space to EC2 instance, that is equal to size of RAM up to 16G
max. Determine the number of blocks by dividing swap size in MB by 128.
swap_blocks: "128"

# Postgresql user configurable parameters
psql_port: "5432"
buffer_cache: "8192MB"
archive_mode: "on"
max_wal_size: "5GB"
client_address: "172.30.15.0/24"
```

Konfigurieren Sie die globale Datei fsx_vars.yml im Ordner „vars“.

```
#############################################################################
######  PostgreSQL HADR global user configuration variables       ######
######  Consolidate all variables from FSx, Linux, and postgresql ######
#############################################################################

##########################################
```

```
### Ontap env specific config variables ###
#########################################

#########################################################################
#########################
# Variables for SnapMirror Peering
#########################################################################
#########################

#Passphrase for cluster peering authentication
passphrase: "xxxxxxx"

#Please enter destination or standby FSx cluster name
dst_cluster_name: "FsxId0cf8e0bccb14805e8"

#Please enter destination or standby FSx cluster management IP
dst_cluster_ip: "172.30.15.90"

#Please enter destination or standby FSx cluster inter-cluster IP
dst_inter_ip: "172.30.15.13"

#Please enter destination or standby SVM name to create mirror
relationship
dst_vserver: "dr"

#Please enter destination or standby SVM management IP
dst_vserver_mgmt_lif: "172.30.15.88"

#Please enter destination or standby SVM NFS lif
dst_nfs_lif: "172.30.15.88"

#Please enter source or primary FSx cluster name
src_cluster_name: "FsxId0cf8e0bccb14805e8"

#Please enter source or primary FSx cluster management IP
src_cluster_ip: "172.30.15.20"

#Please enter source or primary FSx cluster inter-cluster IP
src_inter_ip: "172.30.15.5"

#Please enter source or primary SVM name to create mirror relationship
src_vserver: "prod"

#Please enter source or primary SVM management IP
src_vserver_mgmt_lif: "172.30.15.115"

#########################################################################
```

```
############################
# Variable for PostgreSQL Volumes, lif - source or primary FSx NFS lif
address
#############################################################################
############################

src_db_vols:
  - {vol_name: "{{groups.postgresql[0]}}_pgdata", aggr_name: "aggr1", lif:
"172.21.94.200", size: "100"}

src_archivelog_vols:
  - {vol_name: "{{groups.postgresql[0]}}_pglogs", aggr_name: "aggr1", lif:
"172.21.94.200", size: "100"}

#Names of the Nodes in the ONTAP Cluster
nfs_export_policy: "default"

#############################################################################
############################
### Linux env specific config variables ###
#############################################################################
############################

#NFS Mount points for PostgreSQL DB volumes
mount_points:
  - "/pgdata"
  - "/pglogs"

#RedHat subscription username and password
redhat_sub_username: "xxxxx"
redhat_sub_password: "xxxxx"

#####################################################
### DB env specific install and config variables ###
#####################################################
#The latest version of PostgreSQL RPM is pulled/installed and config file
is deployed from a preconfigured template
#Recovery type and point: default as all logs and promote and leave all
PITR parameters blank
```

PostgreSQL-Bereitstellung und HA/DR-Setup

Die folgenden Aufgaben stellen den PostgreSQL-DB-Serverdienst bereit und initialisieren die Datenbank am primären Standort auf dem primären EC2-DB-Serverhost. Anschließend wird am Standby-Standort ein primärer EC2-DB-Server-Host als Standby eingerichtet. Schließlich wird die DB-Volume-Replikation vom FSx-Cluster des primären Standorts zum FSx-Cluster des Standby-Standorts für die Notfallwiederherstellung eingerichtet.

1. Erstellen Sie DB-Volumes auf dem primären FSx-Cluster und richten Sie PostgreSQL auf dem primären EC2-Instance-Host ein.

```
ansible-playbook -i hosts postgresql_deploy.yml -u ec2-user --private
-key psql_01p.pem -e @vars/fsx_vars.yml
```

2. Richten Sie den Standby-DR-EC2-Instance-Host ein.

```
ansible-playbook -i hosts postgresql_standby_setup.yml -u ec2-user
--private-key psql_01s.pem -e @vars/fsx_vars.yml
```

3. Richten Sie FSx ONTAP Cluster-Peering und Datenbank-Volume-Replikation ein.

```
ansible-playbook -i hosts fsx_replication_setup.yml -e
@vars/fsx_vars.yml
```

4. Konsolidieren Sie die vorherigen Schritte in einer einstufigen PostgreSQL-Bereitstellung und HA/DR-Einrichtung.

```
ansible-playbook -i hosts postgresql_hadr_setup.yml -u ec2-user -e
@vars/fsx_vars.yml
```

5. Um einen Standby-PostgreSQL-DB-Host entweder am primären oder am Standby-Standort einzurichten, kommentieren Sie alle anderen Server im Abschnitt [dr_postgresql] der Hosts-Datei aus und führen Sie dann das Playbook postgresql_standby_setup.yml mit dem jeweiligen Zielhost aus (z. B. psql_01ps oder Standby-EC2-Compute-Instanz am primären Standort). Stellen Sie sicher, dass eine Hostparameterdatei wie `psql_01ps.yml` wird unter dem `host_vars` Verzeichnis.

```
[dr_postgresql] --
#psql_01s ansible_ssh_private_key_file=psql_01s.pem
psql_01ps ansible_ssh_private_key_file=psql_01ps.pem
#psql_01ss ansible_ssh_private_key_file=psql_01ss.pem
```

```
ansible-playbook -i hosts postgresql_standby_setup.yml -u ec2-user
--private-key psql_01ps.pem -e @vars/fsx_vars.yml
```

Sicherung und Replikation von PostgreSQL-Datenbank-Snapshots auf Standby-Site

Die Sicherung und Replikation von Snapshots der PostgreSQL-Datenbank auf die Standby-Site kann auf dem Ansible-Controller in einem benutzerdefinierten Intervall gesteuert und ausgeführt werden. Wir haben bestätigt,

dass das Intervall nur 5 Minuten betragen kann. Daher besteht im Falle eines Fehlers am primären Standort ein potenzieller Datenverlust von 5 Minuten, wenn der Fehler direkt vor der nächsten geplanten Snapshot-Sicherung auftritt.

```
*/15 * * * * /home/admin/na_postgresql_aws_deploy_hadr/data_log_snap.sh
```

Failover zum Standby-Standort für DR

Um das PostgreSQL HA/DR-System als DR-Übung zu testen, führen Sie ein Failover und eine PostgreSQL-Datenbankwiederherstellung auf der primären Standby-EC2-DB-Instance auf der Standby-Site aus, indem Sie das folgende Playbook ausführen. Führen Sie in einem tatsächlichen DR-Szenario dasselbe für ein tatsächliches Failover zur DR-Site aus.

```
ansible-playbook -i hosts postgresql_failover.yml -u ec2-user --private
-key psql_01s.pem -e @vars/fsx_vars.yml
```

Resynchronisieren Sie replizierte DB-Volumes nach dem Failover-Test

Führen Sie nach dem Failover-Test eine erneute Synchronisierung aus, um die SnapMirror -Replikation des Datenbank-Volumes wiederherzustellen.

```
ansible-playbook -i hosts postgresql_standby_resync.yml -u ec2-user
--private-key psql_01s.pem -e @vars/fsx_vars.yml
```

Failover vom primären EC2-DB-Server zum Standby-EC2-DB-Server aufgrund eines Fehlers der EC2-Compute-Instanz

NetApp empfiehlt die Ausführung eines manuellen Failovers oder die Verwendung etablierter OS-Clusterware, für die möglicherweise eine Lizenz erforderlich ist.

Wo Sie weitere Informationen finden

Weitere Informationen zu den in diesem Dokument beschriebenen Informationen finden Sie in den folgenden Dokumenten und/oder auf den folgenden Websites:

- Amazon FSx ONTAP

"https://aws.amazon.com/fsx/netapp-ontap/"

- Amazon EC2

https://aws.amazon.com/pm/ec2/?trk=36c6da98-7b20-48fa-8225-4784bced9843&sc_channel=ps&s_kwcid=AL!4422!3!467723097970!e!!g!!aws%20ec2&ef_id=Cj0KCQiA54KfBhCKARIsAJzSrdqwQrghn6I71jiWzSeaT9Uh1-vY-VfhJixF-xnv5rWwn2S7RqZOTQ0aAh7eEALw_wcB:G:s&s_kwcid=AL!4422!3!467723097970!e!!g!!aws%20ec2

- NetApp Lösungsautomatisierung

"Einführung"