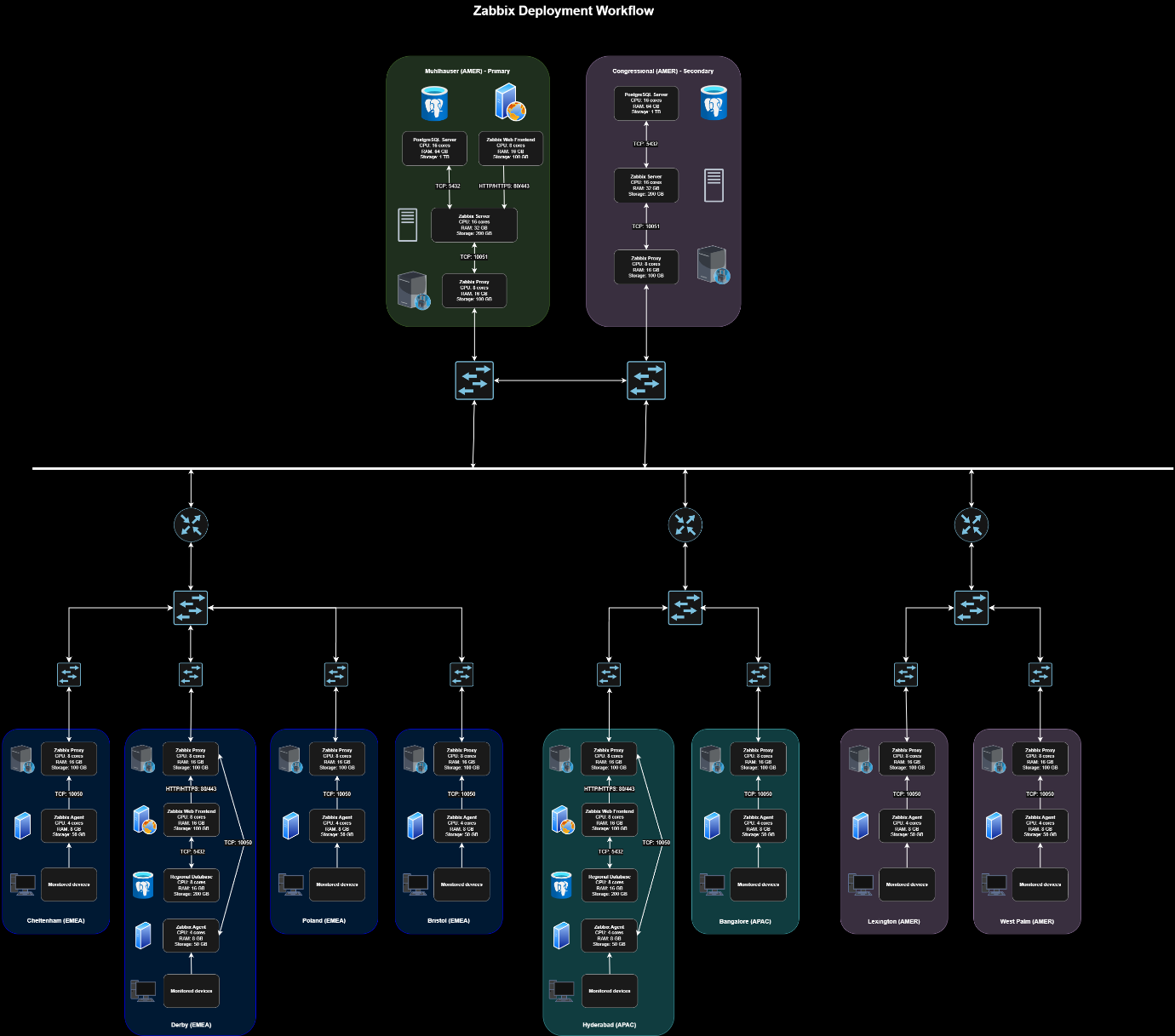
**Zabbix Deployment Workflow**



Below is a step-by-step installation guide tailored for a Zabbix deployment on Rocky Linux 9 based on the provided Zabbix Deployment Workflow diagram. This setup distributes the workload across separate servers for better performance and scalability. The guide will use Zabbix 7.0 LTS on Rocky Linux 9, with NGINX as the web frontend and PostgreSQL as the database, while aligning with the regional structure.

The workflow involves four dedicated server types:

* **PostgreSQL Server**: Hosts the Zabbix database (primary and secondary).
* **Zabbix Server**: Runs the Zabbix server process (primary and secondary for HA).
* **NGINX Web Frontend Server**: Hosts the Zabbix web interface.
* **Zabbix Proxy Server**: Manages proxies for each region

The components are split across dedicated servers. The Primary and Secondary Zabbix Servers will connect to their respective PostgreSQL servers, the NGINX server will serve the web frontend, and each region’s proxy will run on a dedicated proxy server.

**Step-by-Step Installation Guide**

**1. Prepare the Infrastructure**

* Ensure all servers (PostgreSQL, Zabbix Server, NGINX Web Frontend, Zabbix Proxy, and Agents) run **Rocky Linux 9**.
* Resource Allocations
  + PostgreSQL Server:
    - CPU: 16 cores
    - RAM: 64GB
    - Storage: 1TB
  + Zabbix Server:
    - CPU: 16 cores
    - RAM: 32 GB
    - Storage: 200 GB
  + NGINX Web Frontend Server
    - CPU: 8 cores
    - RAM: 16 GB
    - Storage: 100 GB
  + Zabbix Proxy Server
    - CPU: 8 cores
    - RAM: 16 GB
    - Storage: 100 GB
* Open required ports using firewalld on each server:

**PostgreSQL Server:** TCP 5432 (PostgreSQL)

**Zabbix Server**: TCP 10051 (server-agent/proxy communication)

**Zabbix Proxy Server**: TCP 10050 (proxy-agent communication)

**NGINX Web Frontend Server**: TCP 80 or 443 (web access)

*# PostgreSQL Server*

*sudo firewall-cmd --add-port=5432/tcp –permanent*

*sudo firewall-cmd --reload*

*# Zabbix Server*

*sudo firewall-cmd --add-port=10051/tcp –permanent*

*sudo firewall-cmd --reload*

*# Zabbix Proxy Server*

*sudo firewall-cmd --add-port=10050/tcp –permanent*

*sudo firewall-cmd –reload*

*# NGINX Web Frontend Server*

*sudo firewall-cmd --add-port=80/tcp –permanent*

*sudo firewall-cmd --add-port=443/tcp –permanent*

*sudo firewall-cmd --reload*

* Ensure network connectivity between servers:
  + Zabbix Server must reach PostgreSQL Server (port 5432).
  + NGINX Server must reach Zabbix Server (port 10051) and PostgreSQL Server (port 5432).
  + Zabbix Proxy Servers must reach Zabbix Server (port 10051).
  + Agents must reach their respective Zabbix Proxy (port 10050).

**2. Install the PostgreSQL Server (Primary and Secondary)**

This server will host the Zabbix database for the Primary and Secondary Zabbix Servers.

* **Install PostgreSQL**

*sudo dnf update*

*sudo dnf install postgresql-server*

* **Verify PostgreSQL Version**

After installation, check the PostgreSQL version to ensure correct file paths:

*psql --version*

This guide assumes PostgreSQL 15

* **Initialize and Start PostgreSQL**

*sudo postgresql-setup initdb*

*sudo systemctl enable postgresql*

*sudo systemctl start postgresql*

* **Configure PostgreSQL**

Create the Zabbix database and user.

*sudo -u postgres psql*

*CREATE DATABASE zabbix;*

*CREATE USER zabbix WITH PASSWORD 'your\_password';*

*GRANT ALL PRIVILEGES ON DATABASE zabbix TO zabbix;*

*\q*

* **Allow Remote Connections**

Edit /var/lib/pgsql/15/data/postgresql.conf

*listen\_addresses = '\*' # Listen on all interfaces*

* Edit /var/lib/pgsql/15/data/pg\_hba.conf to allow connections from Zabbix Server and NGINX Server (replace <zabbix\_server\_ip> and <nginx\_server\_ip> with actual IPs):

*host zabbix zabbix <zabbix\_server\_ip>/32 md5*

*host zabbix zabbix <nginx\_server\_ip>/32 md5*

* **Restart PostgreSQL**

*sudo systemctl restart postgresql*

* **Repeat for Secondary PostgreSQL Server**
  + Follow the same steps on the secondary PostgreSQL server.
  + Set up replication between the primary and secondary PostgreSQL servers. (See the replication guide in the document)
  + In pg\_hba.conf on the primary, add:

*host replication zabbix\_repl <secondary\_postgresql\_ip>/32 md5*

**3. Install the Primary Zabbix Server**

* **Install Dependencies**

*sudo dnf update*

*sudo dnf install epel-release*

* **Add Zabbix Repository**

*sudo rpm -Uvh https://repo.zabbix.com/zabbix/7.0/rhel/9/x86\_64/zabbix-release-7.0-4.el9.noarch.rpm*

*sudo dnf clean all*

*sudo dnf makecache*

* **Install Zabbix Server and Agent**

*sudo dnf install zabbix-server-pgsql zabbix-agent zabbix-sql-scripts*

* **Import Schema**

Ensure psql is installed (sudo dnf install postgresql for the client). Run the schema import, specifying the PostgreSQL server’s IP:

*sudo zcat /usr/share/zabbix-sql-scripts/postgresql/server.sql.gz | psql -h <postgresql\_server\_ip> -U zabbix -d zabbix*

Enter the password (your\_password) when prompted.

* **Configure Zabbix Server**

Edit /etc/zabbix/zabbix\_server.conf

*DBHost=<postgresql\_server\_ip>*

*DBName=zabbix*

*DBUser=zabbix*

*DBPassword=your\_password*

* **Start Services**

*sudo systemctl enable zabbix-server zabbix-agent*

*sudo systemctl start zabbix-server zabbix-agent*

**4. Install the Secondary Zabbix Server**

* Repeat the Primary Zabbix Server steps on the secondary Zabbix Server.
* Configure it to connect to the secondary PostgreSQL server (set up with replication).
* Update /etc/zabbix/zabbix\_server.conf for HA:

*DBHost=<secondary\_postgresql\_server\_ip>*

*DBName=zabbix*

*DBUser=zabbix*

*DBPassword=your\_password*

*HANodeName=zbx-secondary*

*NodeAddress=<secondary\_zabbix\_server\_ip>:10051*

**5. Install the NGINX Web Frontend Server**

This server will host the Zabbix web interface and connect to the Primary Zabbix Server and PostgreSQL Server.

* **Install Dependencies**

*sudo dnf update*

*sudo dnf install epel-release*

*sudo dnf install nginx php php-fpm php-pgsql*

* **Add Zabbix Repository**

*sudo rpm -Uvh https://repo.zabbix.com/zabbix/7.0/rhel/9/x86\_64/zabbix-release-7.0-4.el9.noarch.rpm*

*sudo dnf clean all*

*sudo dnf makecache*

* **Install Zabbix Web Frontend**

*sudo dnf install zabbix-web-pgsql zabbix-nginx-conf*

* **Configure PHP**

Edit /etc/php.ini

*date.timezone = UTC*

* **Configure NGINX**

Edit /etc/nginx/conf.d/zabbix.conf

*server {*

*listen 80;*

*server\_name <nginx\_server\_ip\_or\_domain>;*

*root /usr/share/zabbix;*

*index index.php;*

*location / {*

*try\_files $uri $uri/ /index.php?$args;*

*}*

*location ~ \.php$ {*

*include fastcgi\_params;*

*fastcgi\_pass unix:/var/run/php-fpm/www.sock;*

*fastcgi\_index index.php;*

*fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;*

*}*

*location ~\* \.(jpg|jpeg|png|gif|ico|css|js)$ {*

*access\_log off;*

*expires 30d;*

*}*

*}*

* **Configure PHP-FPM**

Edit /etc/php-fpm.d/www.conf

*listen = /var/run/php-fpm/www.sock*

*listen.owner = nginx*

*listen.group = nginx*

*listen.mode = 0660*

* **Configure Zabbix Frontend**

Edit /etc/zabbix/web/zabbix.conf.php to point to the PostgreSQL Server and Zabbix Server:

*$DB['TYPE'] = 'POSTGRESQL';*

*$DB['SERVER'] = '<postgresql\_server\_ip>';*

*$DB['PORT'] = '5432';*

*$DB['DATABASE'] = 'zabbix';*

*$DB['USER'] = 'zabbix';*

*$DB['PASSWORD'] = 'your\_password';*

*$ZBX\_SERVER = '<primary\_zabbix\_server\_ip>';*

*$ZBX\_SERVER\_PORT = '10051';*

*$ZBX\_SERVER\_NAME = 'Zabbix';*

* **Set Permissions**

*sudo chown -R zabbix:nginx /usr/share/zabbix*

*sudo chown -R nginx:nginx /var/lib/php/session*

* **Start Services**

*sudo systemctl enable nginx php-fpm*

*sudo systemctl start nginx php-fpm*

**6. Install Zabbix Proxies**

* **Install Proxy**

*sudo dnf update*

*sudo dnf install epel-release*

*sudo rpm -Uvh https://repo.zabbix.com/zabbix/7.0/rhel/9/x86\_64/zabbix-release-7.0-4.el9.noarch.rpm*

*sudo dnf clean all*

*sudo dnf makecache*

*sudo dnf install zabbix-proxy-pgsql postgresql-server*

* **Initialize and Start PostgreSQL (local to the proxy server):**

*sudo postgresql-setup initdb*

*sudo systemctl enable postgresql*

*sudo systemctl start postgresql*

* **Configure Database**

sudo -u postgres psql

CREATE DATABASE zabbix\_proxy\_<region>;

CREATE USER zabbix\_proxy\_<region> WITH PASSWORD 'your\_password';

GRANT ALL PRIVILEGES ON DATABASE zabbix\_proxy\_<region> TO zabbix\_proxy\_<region>;

\q

* **Import Schema**

*sudo zcat /usr/share/zabbix-sql-scripts/postgresql/schema.sql.gz | sudo -u zabbix\_proxy\_<region> psql zabbix\_proxy\_<region>*

* **Configure Proxy**

Edit /etc/zabbix/zabbix\_proxy.conf

*Server=<primary\_zabbix\_server\_ip>*

*DBName=zabbix\_proxy\_<region>*

*DBUser=zabbix\_proxy\_<region>*

*DBPassword=your\_password*

*ProxyMode=0 # Active proxy*

*Hostname=Zabbix Proxy <region>*

* **Start Proxy**

*sudo systemctl enable zabbix-proxy postgresql*

*sudo systemctl start zabbix-proxy postgresql*

Repeat for each region.

**7. Install Zabbix Agents**

Install agents on all monitored devices under each proxy

* **Install Agent**

*sudo dnf install zabbix-agent*

* **Configure Agent**

Edit /etc/zabbix/zabbix\_agentd.conf

*Server=<proxy\_server\_ip\_for\_region>*

*ServerActive=<proxy\_server\_ip\_for\_region>*

*Hostname=<device\_name>*

* **Start Agent**

*sudo systemctl enable zabbix-agent*

*sudo systemctl start zabbix-agent*

**8. Configure Communication**

Verify connectivity:

* Zabbix Server to PostgreSQL Server (port 5432).
* NGINX Server to Zabbix Server (port 10051) and PostgreSQL Server (port 5432).
* Zabbix Proxy to Zabbix Server (port 10051).
* Agents to Zabbix Proxy (port 10050).

Use zabbix\_get or netstat to test.

* Add proxies and hosts in the Zabbix web interface (http://<nginx\_server\_ip>/zabbix):
* Configuration > Proxies > Create Proxy: Add each proxy with its respective server IP.
* Configuration > Hosts > Create Host: Add devices with proxy assignments.

**9. Test and Validate**

* Log in to the Zabbix frontend (http://<nginx\_server\_ip>/zabbix, default: Admin/zabbix).
* Check statuses in Monitoring > Hosts and Monitoring > Proxies.
* Verify data collection from all regions in Monitoring > Latest Data.

**10. Finalize Deployment**

* Configure monitoring templates, triggers, and alerts via the web interface.
* Back up configurations and databases:
  + PostgreSQL Server: Back up the database (pg\_dump).
  + Zabbix Server: Back up /etc/zabbix/.
  + NGINX Server: Back up /etc/nginx/ and /etc/zabbix/web/.
  + Proxy Servers: Back up /etc/zabbix/ and local PostgreSQL databases.
* Monitor the HA Setup

Configure Zabbix to monitor PostgreSQL server in the Zabbix Frontend

* + Create a host for each PostgreSQL server in the Zabbix Frontend.
  + Use the PostgreSQL template to monitor replication (pg\_stat\_replication metrics).

Monitor Zabbix Server HA

* + Add a trigger to alert if the Primary Zabbix Server is down and the Secondary has not taken over.

Monitor NGINX and Proxy servers for availability and performance.

* Secure the Setup
  + Change default passwords for all components.
  + Enable HTTPS on the NGINX server
  + Restrict database access in pg\_hba.conf on PostgreSQL servers.
  + Secure NGINX by disabling directory listing and restricting access to sensitive files.
  + Secure Zabbix Communication with PSK
    - Generate a PSK for each proxy and agent:

*openssl rand -hex 32 > /etc/zabbix/zabbix\_proxy.psk*

* + - Edit /etc/zabbix/zabbix\_proxy.conf on each proxy:

*TLSConnect=psk*

*TLSAccept=psk*

*TLSPSKIdentity=ZabbixProxy\_<region>*

*TLSPSKFile=/etc/zabbix/zabbix\_proxy.psk*

* + - Edit /etc/zabbix/zabbix\_server.conf on the Zabbix Server to accept PSK connections:

*TLSConnect=psk*

*TLSAccept=psk*

* + - Configure the PSK in the Zabbix frontend (Configuration > Proxies > Encryption).
    - Repeat for agents by editing /etc/zabbix/zabbix\_agentd.conf and configuring PSK in the frontend.

**References**:

Zabbix 7.0 Documentation: <https://www.zabbix.com/documentation/7.0/>

PostgreSQL Replication: <https://www.postgresql.org/docs/current/high-availability.html>

NGINX Configuration: <https://www.nginx.com/resources/wiki/>

**PostgreSQL Replication for Secondary Server**

Here is the guide for setting up PostgreSQL replication between the dedicated Primary PostgreSQL Server and Secondary PostgreSQL Server, ensuring compatibility with distributed Zabbix setup.

**Prerequisites**

* Two dedicated PostgreSQL servers:
  + Primary PostgreSQL Server (master)
  + Secondary PostgreSQL Server (standby)
* Both servers have PostgreSQL installed and the Zabbix database (zabbix) created (as per the previous steps in the dedicated servers guide).
* Network connectivity between the two PostgreSQL servers:
  + Port 5432 (PostgreSQL) open for replication.
  + Port 22 (SSH) open for rsync.
* rsync and SSH configured for file transfer between servers (preferably with key-based authentication for automation).
* Root or sudo access on both servers.
* The Primary and Secondary Zabbix Servers (running on separate dedicated servers) are configured to connect to their respective PostgreSQL servers.

**Step-by-Step Guide to Configure PostgreSQL Replication**

**1. Prepare the Primary PostgreSQL Server**

* **Stop PostgreSQL**

Ensure no changes occur during the initial setup.

*sudo systemctl stop postgresql*

* **Backup the Data Directory**

Create a backup of the current PostgreSQL data directory

*sudo cp -r /var/lib/pgsql/15/data/ /var/lib/pgsql/15/data\_backup/*

* **Configure the primary server for replication**

Edit /var/lib/pgsql/15/data/postgresql.conf and set the following parameters:

*wal\_level = replica*

*max\_wal\_senders = 10*

*wal\_keep\_size = 128MB*

*hot\_standby = off # Enable on standby later*

*archive\_mode = on # Added for WAL archiving*

*archive\_command = 'cp %p /var/lib/pgsql/15/archive/%f'*

Create the archive directory

*sudo mkdir -p /var/lib/pgsql/15/archive*

*sudo chown postgres:postgres /var/lib/pgsql/15/archive*

* **Allow replication connections**

Edit /var/lib/pgsql/15/data/pg\_hba.conf & add the following line to allow replication from the secondary PostgreSQL server (replace <secondary\_postgresql\_ip> with the secondary PostgreSQL server’s IP):

*host replication zabbix\_repl <secondary\_postgresql\_ip>/32 md5*

* **Restart PostgreSQL**

*sudo systemctl start postgresql*

* **Create a role for replication**

*sudo -u postgres psql*

*CREATE ROLE zabbix\_repl WITH REPLICATION LOGIN PASSWORD 'your\_repl\_password';*

*\q*

**2. Prepare the Secondary PostgreSQL Server**

* **Stop PostgreSQL**

*sudo systemctl stop postgresql*

* **Remove Existing Data**

Clear the existing data directory to avoid conflicts.

*sudo rm -rf /var/lib/pgsql/15/data/\**

* **Sync Data from Primary**

Use rsync to copy the primary PostgreSQL server’s data directory to the secondary. On the secondary PostgreSQL server, run:

*sudo -u postgres rsync -av -e ssh postgres@<primary\_postgresql\_ip>:/var/lib/pgsql/15/data/\* /var/lib/pgsql/15/data/*

Ensure SSH is set up with key-based authentication for automation.

* **Configure the Standby Server**

Edit /var/lib/pgsql/15/data/postgresql.conf

*hot\_standby = on*

*primary\_conninfo = 'host=<primary\_postgresql\_ip> port=5432 user=zabbix\_repl password=your\_repl\_password'*

*promote\_trigger\_file = '/tmp/postgresql.trigger'*

*recovery\_target\_timeline = 'latest'*

*restore\_command = 'cp /var/lib/pgsql/15/archive/%f %p' # Added for WAL archiving*

Create archive directory

*sudo mkdir -p /var/lib/pgsql/15/archive*

*sudo chown postgres:postgres /var/lib/pgsql/15/archive*

Create an empty standby.signal file:

*sudo -u postgres touch /var/lib/pgsql/15/data/standby.signal*

* **Set Permissions**

*sudo chown -R postgres:postgres /var/lib/pgsql/15/data*

3. **Start the Secondary PostgreSQL Server**

* **Start PostgreSQL**

*sudo systemctl start postgresql*

* **Verify recovery mode**

Check if the secondary server is in recovery mode (indicating it’s acting as a standby)

*sudo -u postgres psql -c "SELECT pg\_is\_in\_recovery();"*

It should return ‘t’ (true) if in recovery mode.

4. Test and Monitor Replication

* **Check Replication Status**

On the primary PostgreSQL server, run:

*sudo -u postgres psql -c "SELECT \* FROM pg\_stat\_replication;"*

This should show the secondary PostgreSQL server’s connection and replication status.

* **Test Failover**
  + Stop the primary PostgreSQL server

*sudo systemctl stop postgresql*

* + On the secondary PostgreSQL server, promote it to primary:

*sudo -u postgres touch /tmp/postgresql.trigger*

*sudo systemctl restart postgresql*

* + Verify the secondary is now the primary:

*sudo -u postgres psql -c "SELECT pg\_is\_in\_recovery();"*

It should return f (false), indicating it is now the primary.

* **Reconfigure After Failover**

After failover, set up the old primary PostgreSQL server as the new standby by reversing the process. Stop the old primary, clear its data directory, rsync from the new primary, and configure it as a standby.

5. **Integrate with Zabbix HA**

* **Configure Zabbix HA**

On the Primary Zabbix Server, edit /etc/zabbix/zabbix\_server.conf

*DBHost=<primary\_postgresql\_ip>*

*DBName=zabbix*

*DBUser=zabbix*

*DBPassword=your\_password*

*HANodeName=zbx-primary*

*NodeAddress=<primary\_zabbix\_server\_ip>:10051*

On the Secondary Zabbix Server, edit /etc/zabbix/zabbix\_server.conf

*DBHost=<secondary\_postgresql\_ip>*

*DBName=zabbix*

*DBUser=zabbix*

*DBPassword=your\_password*

*HANodeName=zbx-secondary*

*NodeAddress=<secondary\_zabbix\_server\_ip>:10051*

* **Update Zabbix Frontend**

On the dedicated NGINX Web Frontend Server, ensure /etc/zabbix/web/zabbix.conf.php can handle failover by pointing to both PostgreSQL servers (you may need a manual reconfiguration after failover):

*$DB['TYPE'] = 'POSTGRESQL';*

*$DB['SERVER'] = '<primary\_postgresql\_ip>'; # Update to secondary IP after failover*

*$DB['PORT'] = '5432';*

*$DB['DATABASE'] = 'zabbix';*

*$DB['USER'] = 'zabbix';*

*$DB['PASSWORD'] = 'your\_password';*

*$ZBX\_SERVER = '<primary\_zabbix\_server\_ip>'; # Update to secondary IP after failover*

*$ZBX\_SERVER\_PORT = '10051';*

*$ZBX\_SERVER\_NAME = 'Zabbix';*

* **Enable HA in the Zabbix frontend**
  + Log in to the Zabbix web interface (http://<nginx\_server\_ip>/zabbix).
  + Navigate to **Administration > General > HA**
  + Enable HA and configure the nodes (Primary and Secondary Zabbix Servers).
* **Test failover**
  + Stop the primary Zabbix server

*sudo systemctl stop zabbix-server*

* + Verify the Secondary Zabbix Server takes over by checking the Zabbix frontend for continuity.

6. **Finalize and Secure**

* **Enable automatic startup on both Primary and Secondary PostgreSQL Servers.**

*sudo systemctl enable postgresql*

* **Regularly Backups**
  + Back up the PostgreSQL WAL files and configuration (/var/lib/pgsql/15/data/postgresql.conf, pg\_hba.conf).
  + Schedule daily full backups with pg\_basebackup using a cron job:

*sudo crontab -u postgres -e*

*0 2 \* \* \* pg\_basebackup -D /path/to/backup -h <primary\_postgresql\_ip> -U zabbix\_repl -P --wal-method=stream*

* + Enable Point-in-Time Recovery (PITR) by archiving WAL files (as configured earlier).
  + Back up Zabbix configuration files daily:

*sudo crontab -e*

*0 3 \* \* \* tar -czf /path/to/backup/zabbix\_configs\_$(date +\%F).tar.gz /etc/zabbix/ /etc/nginx/ /usr/share/zabbix/*

* + Store backups in a secure, off-site location.
* **Secure the Setup**
  + Use strong passwords for the zabbix\_repl user.
  + Restrict access in pg\_hba.conf to only allow connections from the secondary PostgreSQL server and Zabbix Servers.
  + Enable SSL for PostgreSQL replication:

Generate SSL certificates for PostgreSQL

*sudo openssl req -new -x509 -days 365 -nodes -out /var/lib/pgsql/15/data/server.crt -keyout /var/lib/pgsql/15/data/server.key*

*sudo chown postgres:postgres /var/lib/pgsql/15/data/server.crt /var/lib/pgsql/15/data/server.key*

*sudo chmod 600 /var/lib/pgsql/15/data/server.crt /var/lib/pgsql/15/data/server.key*

Edit /var/lib/pgsql/15/data/postgresql.conf:

*ssl = on*

*ssl\_cert\_file = 'server.crt'*

*ssl\_key\_file = 'server.key'*

Update primary\_conninfo on the secondary to use SSL:

*primary\_conninfo = 'host=<primary\_postgresql\_ip> port=5432 user=zabbix\_repl password=your\_repl\_password sslmode=verify-full'*

Restart PostgreSQL on both servers

**Notes**

1. **Zabbix Server Configuration**:

The Zabbix Servers connect to the PostgreSQL servers via DBHost in zabbix\_server.conf. After a PostgreSQL failover, you may need to update the DBHost to point to the new primary PostgreSQL server, or use a virtual IP/load balancer for seamless failover.

1. **Failover Automation**:

To automate failover, consider using tools like repmgr or Patroni for PostgreSQL, which can handle automatic promotion and failover. The manual trigger\_file method works but requires intervention.

For production environments, consider using repmgr to automate PostgreSQL failover. Install repmgr on both PostgreSQL servers:

*sudo dnf install repmgr*

Configure repmgr to manage replication and failover (refer to the repmgr documentation: https://repmgr.org/docs/). This allows automatic promotion of the standby server and reduces downtime.

1. **Security**:

Ensure communication between PostgreSQL servers is secure (use SSH keys for rsync, enable SSL for replication).

1. **Zabbix Frontend Failover**:

The Zabbix frontend configuration (zabbix.conf.php) may need manual updating after a PostgreSQL failover unless you use a database connection failover mechanism (e.g., a proxy like PgBouncer or a virtual IP).

To avoid manual updates to /etc/zabbix/web/zabbix.conf.php after failover, use a virtual IP (VIP) or load balancer:

Set up Keepalived or HAProxy to manage a VIP that points to the active PostgreSQL and Zabbix Servers.

Configure $DB['SERVER'] and $ZBX\_SERVER in zabbix.conf.php to use the VIP instead of specific IPs.

**References**

Zabbix HA Documentation:

<https://www.zabbix.com/documentation/current/en/manual/installation/high_availability>

PostgreSQL Replication Docs:

<https://www.postgresql.org/docs/current/high-availability.html>

NGINX Configuration:

<https://www.nginx.com/resources/wiki/>