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 covers the Primary Zabbix Server, Secondary Zabbix Server, Proxies, and Agents across the specified regions.

**Step-by-Step Installation Guide**

**1. Prepare the Infrastructure**

* Ensure all servers (Primary Zabbix Server, Secondary Zabbix Server, Proxies, Agents) run Rocky Linux 9.
* Open required ports (e.g., TCP 10051 for server-agent, TCP 10050 for proxy-agent) using firewalld:

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

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

*sudo firewall-cmd --reload*

* Allocate storage (e.g., 100 GB for servers, 10 GB for proxies).

**2. Install the Primary Zabbix Server**

* Install Dependencies: Update the system and install required packages.

*sudo dnf update*

*sudo dnf install epel-release*

*sudo dnf install httpd php php-pgsql postgresql-server*

* Initialize and Start PostgreSQL:

*sudo postgresql-setup initdb*

*sudo systemctl enable postgresql*

*sudo systemctl start postgresql*

* Configure PostgreSQL:

*sudo -u postgres psql*

*CREATE DATABASE zabbix;*

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

*GRANT ALL PRIVILEGES ON DATABASE zabbix TO zabbix;*

*\q*

* Add Zabbix Repository: Install the Zabbix 6.0 repository for Rocky Linux 9.

*sudo rpm -Uvh https://repo.zabbix.com/zabbix/6.0/rhel/9/x86\_64/zabbix-release-6.0-1.el9.noarch.rpm*

*sudo dnf clean all*

*sudo dnf makecache*

* Install Zabbix Server: Install server, frontend, and agent with PostgreSQL support.

*sudo dnf install zabbix-server-pgsql zabbix-web-pgsql zabbix-apache-conf zabbix-agent*

* Import Schema: Import the initial schema into PostgreSQL.

*sudo zcat /usr/share/doc/zabbix-server-pgsql\*/create.sql.gz | sudo -u zabbix psql zabbix*

* Configure Zabbix Server: Edit `/etc/zabbix/zabbix\_server.conf` with database details.

*DBName=zabbix*

*DBUser=zabbix*

*DBPassword=your\_password*

* Configure PHP: Edit `/etc/php.ini` to set `date.timezone` (e.g., `date.timezone = UTC`).
* Start Services:

*sudo systemctl enable httpd zabbix-server zabbix-agent*

*sudo systemctl start httpd zabbix-server zabbix-agent*

**3. Install the Secondary Zabbix Server**

* Repeat the Primary Server steps on the secondary server.
* Set up PostgreSQL replication or configure it as a standby server (consult Zabbix HA docs).
* Update `/etc/zabbix/zabbix\_server.conf` to sync with the primary server.

**4. Install Zabbix Proxies**

* Install proxies for all regions Cheltenham, Poland, Bristol, Bangalore, Lexington, and West Palm.
* Install Proxy:

*sudo dnf install zabbix-proxy-pgsql*

* Configure Database: Set up a local PostgreSQL database for each proxy.

*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/doc/zabbix-proxy-pgsql\*/schema.sql.gz | sudo -u postgres psql zabbix\_proxy\_<region>*

* Configure Proxy: Edit `/etc/zabbix/zabbix\_proxy.conf` with proxy name, server IP, and database details.

*Server=<primary\_server\_ip>*

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

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

*DBPassword=your\_password*

* Start Proxy:

*sudo systemctl enable zabbix-proxy*

*sudo systemctl start zabbix-proxy*

**5. 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` to point to the respective proxy IP.

*Server=<proxy\_ip>*

*ServerActive=<proxy\_ip>*

* Start Agent:

*sudo systemctl enable zabbix-agent*

*sudo systemctl start zabbix-agent*

**6. Configure Communication**

* Verify connectivity between primary server, secondary server, proxies, and agents using `zabbix\_get` or `netstat`.
* Add proxies and monitored hosts in the Zabbix web interface (`http://<server\_ip>/zabbix`).

**7. Test and Validate**

* Log in to the Zabbix frontend (default: Admin/zabbix).
* Check server, proxy, and agent status in the "Monitoring" section.
* Ensure data flows from all regions.

**8. Finalize Deployment**

* Configure monitoring, triggers, and alerts via the web interface.
* Back up PostgreSQL databases and configuration files.

Setting up PostgreSQL replication to configure the Secondary Zabbix Server as a standby server involves configuring streaming replication with a hot standby setup. This ensures high availability (HA) for the Zabbix database, aligning with the Zabbix HA documentation recommendations. Below are detailed steps for Rocky Linux 9 using PostgreSQL, based on the Zabbix Deployment Workflow.

**Prerequisites**

Two servers: Primary Zabbix Server (master) and Secondary Zabbix Server (standby).

Both servers have PostgreSQL installed and the Zabbix database (zabbix) created (as per the previous steps).

Network connectivity between servers with ports 5432 (PostgreSQL) and 22 (SSH) open.

rsync and ssh configured for file transfer between servers.

root or sudo access on both servers.

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

1. Prepare the Primary Server

Stop the Primary Server: 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

default: /var/lib/pgsql/15/data/

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

Edit postgresql.conf: 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

Edit pg\_hba.conf: Allow replication connections from the standby server.

Edit /var/lib/pgsql/15/data/pg\_hba.conf & add the following line (replace `<standby\_ip>` with the standby server's IP):

host replication zabbix\_repl <standby\_ip>/32 md5

Restart PostgreSQL: Apply the changes.

*sudo systemctl start postgresql*

Create Replication User: Create a role for replication.

*sudo -u postgres psql*

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

*\q*

2. Prepare the Standby Server

Stop PostgreSQL: Ensure the standby server is stopped.

*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’s data directory to the standby.

On the standby server, run:

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

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

Edit postgresql.conf on Standby: Configure the standby.

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

hot\_standby = on

Create recovery.conf: Configure the standby to follow the primary.

Create `/var/lib/pgsql/15/data/recovery.conf` (if using PostgreSQL 15, this may be part of `postgresql.conf` with `primary\_conninfo`).

standby\_mode = 'on'

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

trigger\_file = '/tmp/postgresql.trigger'

Save and exit. (Note: In PostgreSQL 12+, `recovery.conf` is deprecated; add `primary\_conninfo` and `standby\_mode` to `postgresql.conf`.)

Set Permissions: Ensure the data directory is owned by the `postgres` user.

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

3. Start the Standby Server

Start the PostgreSQL service on the standby server.

*sudo systemctl start postgresql*

Verify the standby is syncing by checking the log or running on the 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, run:

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

This should show the standby server’s connection.

Test Failover: Stop the primary server and confirm the standby takes over (manual promotion required).

On standby, promote it to primary:

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

*sudo systemctl restart postgresql*

Reconfigure if Needed: After failover, set up the old primary as the new standby by reversing the process.

5. Integrate with Zabbix HA

Update the Zabbix server configuration (`/etc/zabbix/zabbix\_server.conf`) on both servers to point to the correct PostgreSQL instance.

Use Zabbix’s HA features (e.g., active/passive setup) by configuring the secondary server to take over if the primary fails. Edit `/etc/zabbix/zabbix\_server.conf` on the secondary to enable HA mode if supported in your Zabbix version.

Test Zabbix failover by stopping the primary Zabbix server and ensuring the secondary connects to the promoted PostgreSQL instance.

6. Finalize and Secure

Enable automatic startup on both servers:

*sudo systemctl enable postgresql*

Regularly back up the PostgreSQL WAL files and configuration.

Consult the [Zabbix HA Documentation] (https://www.zabbix.com/documentation/current/en/manual/installation/high\_availability) and [PostgreSQL Replication Docs] (https://www.postgresql.org/docs/current/high-availability.html) for advanced tuning (e.g., synchronous replication).

This setup ensures the Secondary Zabbix Server operates as a hot standby, providing HA for the Zabbix database. Let me know if you need further clarification or assistance!