**CẤU HÌNH TRÊN 3 NODE:**

**Cấu hình hosts trên 3 node:**

vi /etc/hosts

10.0.0.1 node1.localdomain node1

10.0.0.2 node2.localdomain node2

10.0.0.3 node3.localdomain node3

**Disable SELINUX trên 3 node:**

vi /etc/selinux/config

SELINUX= disabled

**Disable firewalld trên 3 node:**

systemctl stop firewalld.service

systemctl disable firewalld.service

**Reboot OS 3 node:**

*init 6*

**Cài đăt REPO trên 3 node:**

sudo dnf install -y epel-release

sudo dnf install -y yum-utils

**Cài đặt Postgresql 15 trên 3 node:**

sudo yum update -y

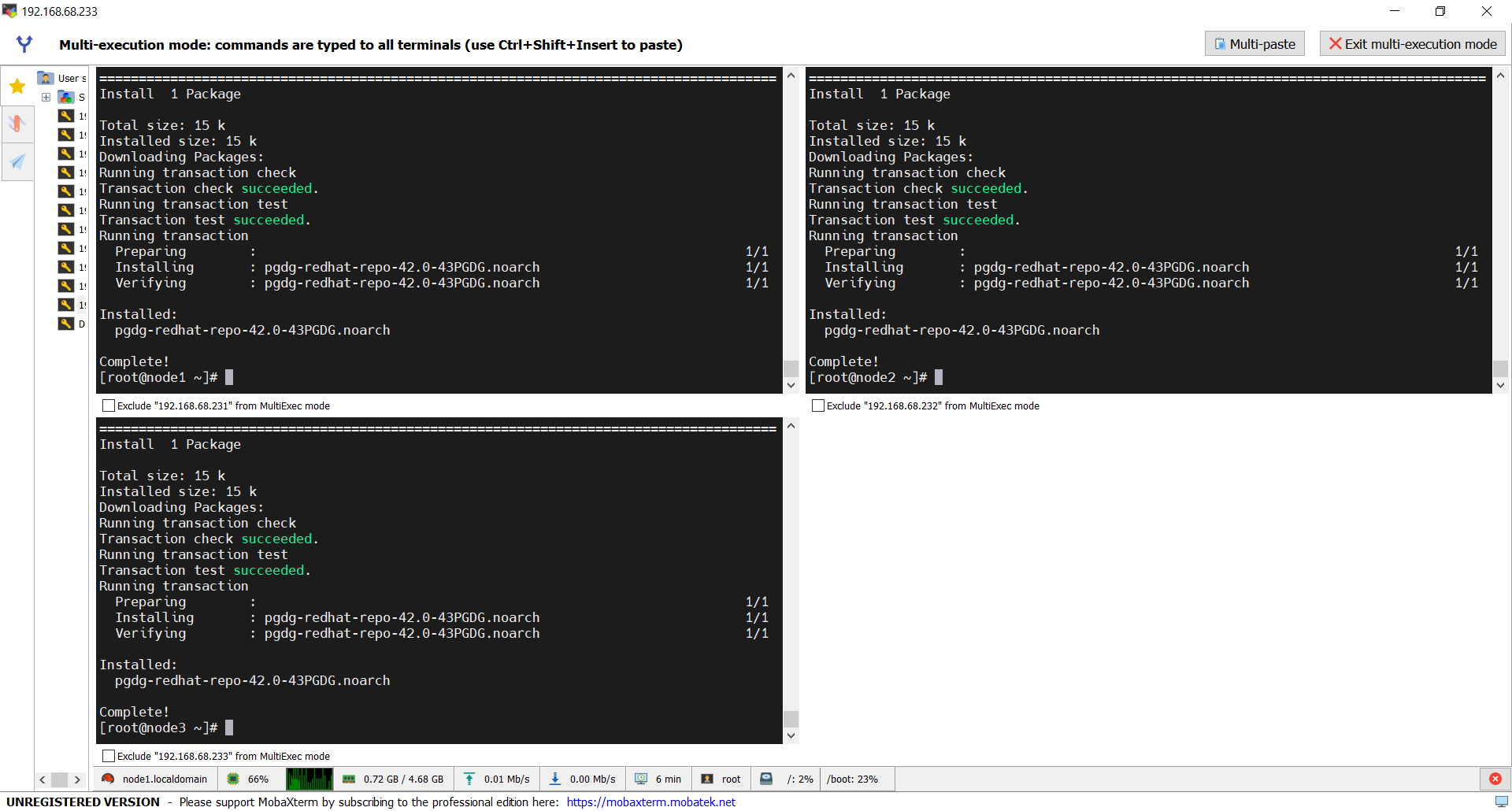
sudo dnf -y install https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86\_64/pgdg-redhat-repo-latest.noarch.rpm

sudo dnf config-manager --enable pgdg15

sudo dnf module disable -y postgresql

sudo dnf -y install postgresql15-server

sudo ln -s /usr/pgsql-15/bin/\* /usr/sbin/



**Cài đặt etcd trên 3 node:**

etcd mặc định là không có trên repo của oracle linux cho nên chúng ta sẽ thực hiện cài đặt thủ công.

sudo vi /etc/yum.repos.d/etcd.repo

## thêm vào file repo

[etcd]

name=PostgreSQL common RPMs for RHEL / oracle $releasever - $basearch

baseurl=http://ftp.postgresql.org/pub/repos/yum/common/pgdg-rhel8-extras/redhat/rhel-$releasever-$basearch

enabled=1

gpgcheck=1

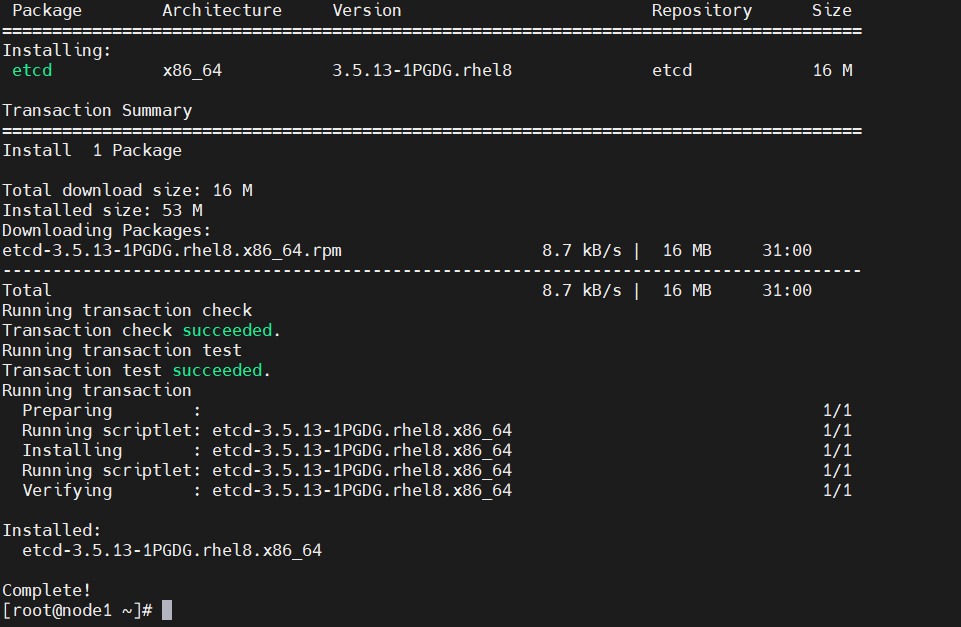
gpgkey=file:///etc/pki/rpm-gpg/PGDG-RPM-GPG-KEY-RHEL

repo\_gpgcheck = 1

## thực hiện cài đặt 3 node:

sudo dnf makecache -y

sudo dnf install -y etcd



*{ ## clear repo cache nếu cần:*

*sudo yum clean all*

*sudo yum repolist*

*}*

**CẤU HÌNH Etcd node node1:**

Sửa thông tin file etcd với nội dung bên dưới, trước đó chúng ta backup lại

sudo mv /etc/etcd/etcd.conf /etc/etcd/etcd.conf.orig

sudo vi /etc/etcd/etcd.conf

## nội dung file:

ETCD\_NAME=node1

ETCD\_DATA\_DIR="/var/lib/etcd/node1"

ETCD\_LISTEN\_PEER\_URLS="http://10.0.0.1:2380"

ETCD\_LISTEN\_CLIENT\_URLS="http://10.0.0.1:2379"

ETCD\_INITIAL\_ADVERTISE\_PEER\_URLS="http://10.0.0.1:2380"

ETCD\_INITIAL\_CLUSTER="node1=http://10.0.0.1:2380,node2=http://10.0.0.2:2380,node3=http://10.0.0.3:2380"

ETCD\_INITIAL\_CLUSTER\_STATE="new"

ETCD\_INITIAL\_CLUSTER\_TOKEN="etcd-cluster"

ETCD\_ADVERTISE\_CLIENT\_URLS="http://10.0.0.1:2379"

ETCD\_ENABLE\_V2="true"

**CẤU HÌNH Etcd node node2:**

Sửa thông tin file etcd với nội dung bên dưới, trước đó chúng ta backup lại

sudo mv /etc/etcd/etcd.conf /etc/etcd/etcd.conf.orig

sudo vi /etc/etcd/etcd.conf

## nội dung file:

ETCD\_NAME=node2

ETCD\_DATA\_DIR="/var/lib/etcd/node2"

ETCD\_LISTEN\_PEER\_URLS="http://10.0.0.2:2380"

ETCD\_LISTEN\_CLIENT\_URLS="http://10.0.0.2:2379"

ETCD\_INITIAL\_ADVERTISE\_PEER\_URLS="http://10.0.0.2:2380"

ETCD\_INITIAL\_CLUSTER="node1=http://10.0.0.1:2380,node2=http://10.0.0.2:2380,node3=http://10.0.0.3:2380"

ETCD\_INITIAL\_CLUSTER\_STATE="new"

ETCD\_INITIAL\_CLUSTER\_TOKEN="etcd-cluster"

ETCD\_ADVERTISE\_CLIENT\_URLS="http://10.0.0.2:2379"

ETCD\_ENABLE\_V2="true"

**CẤU HÌNH Etcd node node3:**

Sửa thông tin file etcd với nội dung bên dưới, trước đó chúng ta backup lại

sudo mv /etc/etcd/etcd.conf /etc/etcd/etcd.conf.orig

sudo vi /etc/etcd/etcd.conf

## nội dung file:

ETCD\_NAME=node3

ETCD\_DATA\_DIR="/var/lib/etcd/node3"

ETCD\_LISTEN\_PEER\_URLS="http://10.0.0.3:2380"

ETCD\_LISTEN\_CLIENT\_URLS="http://10.0.0.3:2379"

ETCD\_INITIAL\_ADVERTISE\_PEER\_URLS="http://10.0.0.3:2380"

ETCD\_INITIAL\_CLUSTER="node1=http://10.0.0.1:2380,node2=http://10.0.0.2:2380,node3=http://10.0.0.3:2380"

ETCD\_INITIAL\_CLUSTER\_STATE="new"

ETCD\_INITIAL\_CLUSTER\_TOKEN="etcd-cluster"

ETCD\_ADVERTISE\_CLIENT\_URLS="http://10.0.0.3:2379"

ETCD\_ENABLE\_V2="true"

**Node1, Node2, Node3:**

sửa thông tin .bash\_profile

su postgres

cd /var/lib/pgsql

vi .bash\_profile

## nội dung file thêm vào bên dưới:

#export PGDATA="/var/lib/pgsql/15/data"

export ETCDCTL\_API="3"

export pgsql0\_ETCD\_URL="http://127.0.0.1:2379"

export pgsql0\_SCOPE="pg\_cluster"

node1=10.0.0.1

node2=10.0.0.2

node3=10.0.0.3

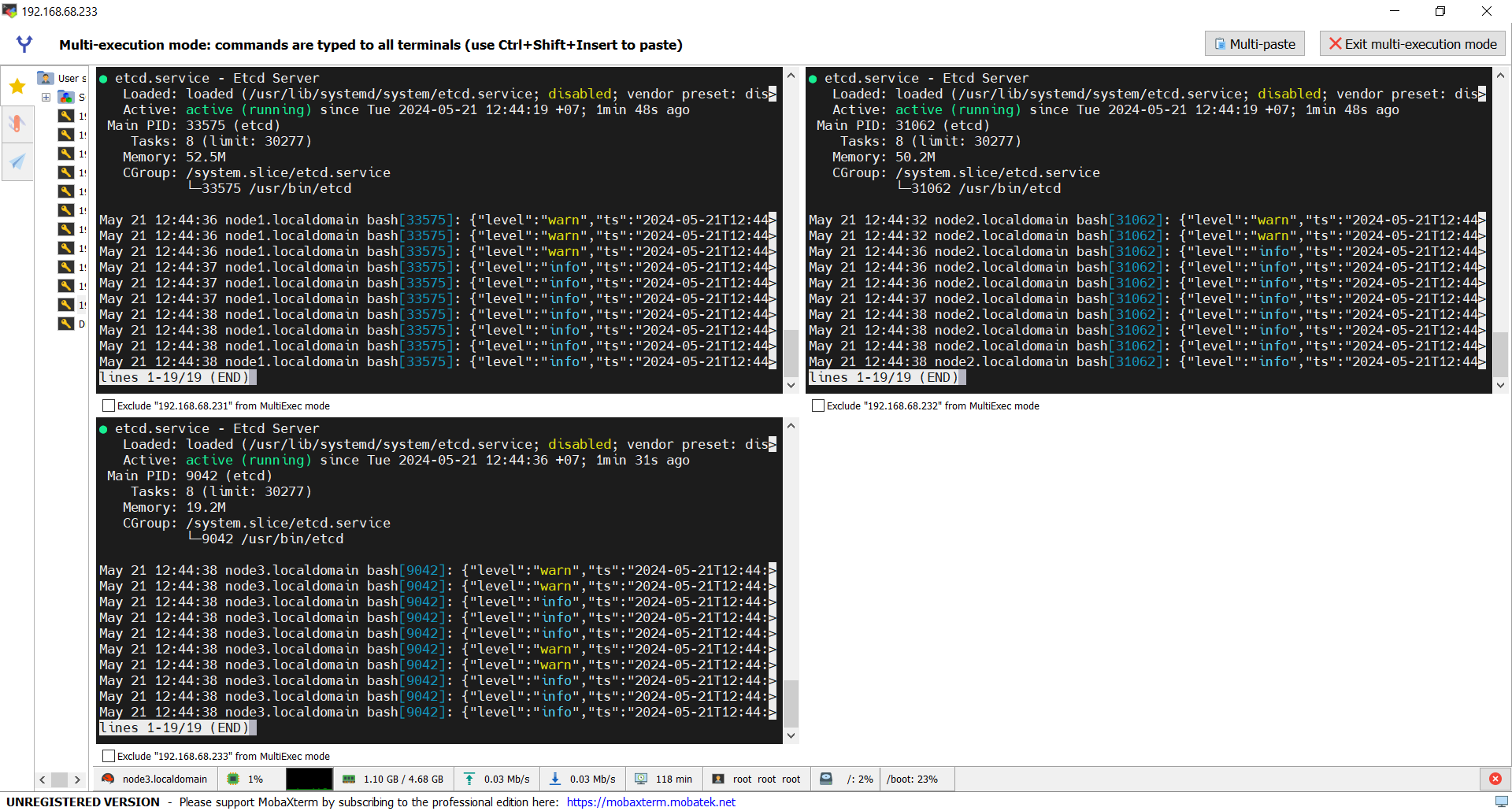
ENDPOINTS=$node1:2379,$node2:2379,$node3:2379

**Node1, Node2, Node3: user ROOT:**

Khởi động etcd Cluster

sudo systemctl start etcd

sudo systemctl status etcd



**Check node (trên node1 hoặc node bất kỳ):**

Chạy .bash\_profile để apply thông tin biến môi trường

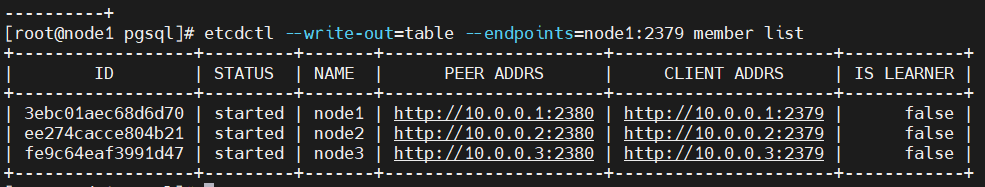
cd /var/lib/pgsql/

. .bash\_profile

Kiểm tra thông tin trạng thái etcd cluster:

etcdctl --write-out=table --endpoints=node1:2379 member list

## kết quả như bên dưới:



**CÀI ĐẶT PATRONI:**

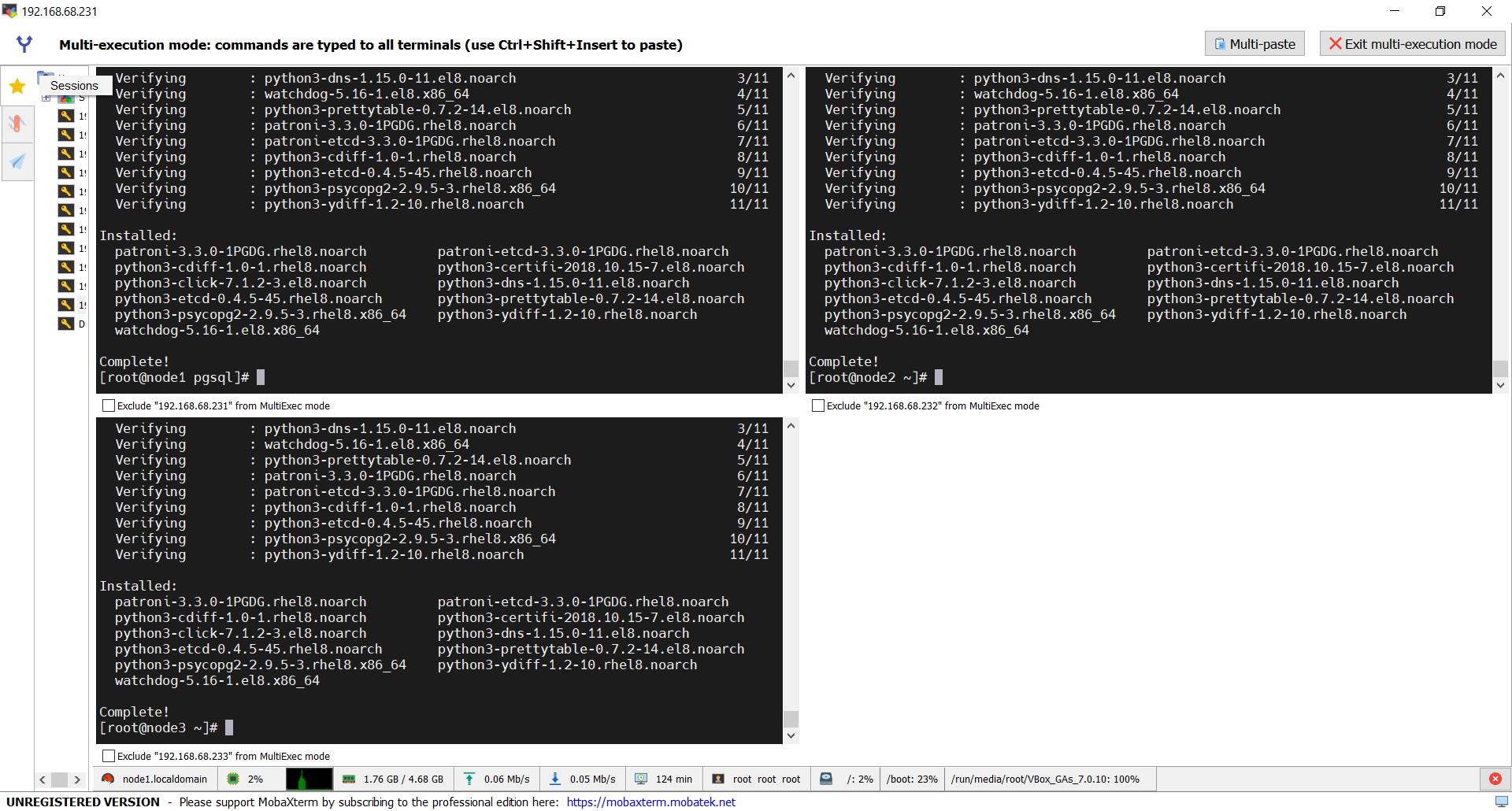
**Node1, Node2, Node3:**

sudo dnf -y install python3 python3-devel python3-pip gcc libpq-devel.x86\_64

sudo pip3 install --upgrade testresources --upgrade setuptools psycopg2 python-etcd

sudo dnf -y install patroni patroni-etcd watchdog

## kết quả cài đặt thành công



**CẤU HÌNH PATRONI CLUSTER**

Theo mặc định, Patroni cấu hình cơ sở dữ liệu PostgreSQL để sao chép không đồng bộ (asynchronous) sử dụng phương pháp sao chép luồng của PostgreSQL. Việc lựa chọn lược đồ sao chép của bạn phụ thuộc vào môi trường production của bạn.

**Sao chép đồng bộ (synchronous) vs. không đồng bộ (asynchronous):**

PostgreSQL hỗ trợ cả sao chép đồng bộ và không đồng bộ để đạt được khả năng sẵn sàng cao và phục hồi thảm họa.

* **Sao chép đồng bộ:** có nghĩa là một hoạt động ghi chỉ được coi là hoàn thành khi nó được xác nhận đã được ghi vào cả máy chủ chính và một hoặc nhiều máy chủ dự phòng đồng bộ. Điều này cung cấp mức độ bền dữ liệu và tính nhất quán cao nhất, vì các ứng dụng khách được đảm bảo rằng các bản ghi của họ đã được sao chép sang ít nhất một máy chủ khác trước khi bản ghi được xác nhận thành công. Tuy nhiên, sự đánh đổi là hiệu suất của máy chủ chính bị ảnh hưởng bởi thời gian cần thiết để xác nhận ghi vào các máy chủ dự phòng đồng bộ, điều này có thể gây ra tắc nghẽn trong môi trường ghi nhiều.
* **Sao chép không đồng bộ:** ngược lại, có nghĩa là một hoạt động ghi được xác nhận thành công ngay khi nó được ghi vào máy chủ chính, mà không cần chờ xác nhận từ các máy chủ dự phòng. Điều này cung cấp hiệu suất tốt hơn vì máy chủ chính không bị cản trở bởi quá trình sao chép, nhưng sự đánh đổi là có nguy cơ mất dữ liệu nếu máy chủ chính bị lỗi trước khi bản ghi được sao chép sang các máy chủ dự phòng.

PostgreSQL cho phép bạn cấu hình một hoặc nhiều bản sao đồng bộ và bất kỳ số lượng bản sao không đồng bộ nào, vì vậy bạn có thể cân bằng giữa hiệu suất và độ bền dữ liệu theo nhu cầu của mình.

Trong bài viết này chúng ta sẽ cấu hình theo asynchronous replication mode Không đồng bộ

**NODE1:**

Chúng ta cần tạo patroni.yml file trong thư mục /etc/patroni

sudo mkdir -p /etc/patroni

sudo vi /etc/patroni/patroni.yml

## nội dung file:

scope: pg\_cluster

namespace: /service/

name: node1

restapi:

listen: 10.0.0.1:8008

connect\_address: 10.0.0.1:8008

etcd:

hosts: 10.0.0.1:2379,10.0.0.2:2379,10.0.0.3:2379

bootstrap:

dcs:

ttl: 30

loop\_wait: 10

retry\_timeout: 10

maximum\_lag\_on\_failover: 1048576

postgresql:

use\_pg\_rewind: true

use\_slots: true

parameters:

initdb:

- encoding: UTF8

- data-checksums

pg\_hba:

- host replication replicator 127.0.0.1/32 md5

- host replication replicator 10.0.0.1/32 md5

- host replication replicator 10.0.0.2/32 md5

- host replication replicator 10.0.0.3/32 md5

- host all all 0.0.0.0/0 md5

users:

admin:

password: admin

options:

- createrole

- createdb

postgresql:

listen: 10.0.0.1:5432

connect\_address: 10.0.0.1:5432

data\_dir: /var/lib/pgsql/15/data

bin\_dir: /usr/pgsql-15/bin

pgpass: /tmp/pgpass

authentication:

replication:

username: replicator

password: replicator

superuser:

username: postgres

password: postgres

watchdog:

mode: required

device: /dev/watchdog

safety\_margin: 5

tags:

nofailover: false

noloadbalance: false

clonefrom: false

nosync: false

**NODE2:**

Chúng ta cần tạo patroni.yml file trong thư mục /etc/patroni

sudo mkdir -p /etc/patroni

sudo vi /etc/patroni/patroni.yml

## nội dung file:

scope: pg\_cluster

namespace: /service/

name: node2

restapi:

listen: 10.0.0.2:8008

connect\_address: 10.0.0.2:8008

etcd:

hosts: 10.0.0.1:2379,10.0.0.2:2379,10.0.0.3:2379

bootstrap:

dcs:

ttl: 30

loop\_wait: 10

retry\_timeout: 10

maximum\_lag\_on\_failover: 1048576

postgresql:

use\_pg\_rewind: true

use\_slots: true

parameters:

initdb:

- encoding: UTF8

- data-checksums

pg\_hba:

- host replication replicator 127.0.0.1/32 md5

- host replication replicator 10.0.0.1/32 md5

- host replication replicator 10.0.0.2/32 md5

- host replication replicator 10.0.0.3/32 md5

- host all all 0.0.0.0/0 md5

users:

admin:

password: admin

options:

- createrole

- createdb

postgresql:

listen: 10.0.0.2:5432

connect\_address: 10.0.0.2:5432

data\_dir: /var/lib/pgsql/15/data

bin\_dir: /usr/pgsql-15/bin

pgpass: /tmp/pgpass

authentication:

replication:

username: replicator

password: replicator

superuser:

username: postgres

password: postgres

watchdog:

mode: required

device: /dev/watchdog

safety\_margin: 5

tags:

nofailover: false

noloadbalance: false

clonefrom: false

nosync: false

**NODE3:**

Chúng ta cần tạo patroni.yml file trong thư mục /etc/patroni

sudo mkdir -p /etc/patroni

sudo vi /etc/patroni/patroni.yml

## nội dung file:

scope: pg\_cluster

namespace: /service/

name: node3

restapi:

listen: 10.0.0.3:8008

connect\_address: 10.0.0.3:8008

etcd:

hosts: 10.0.0.1:2379,10.0.0.2:2379,10.0.0.3:2379

bootstrap:

dcs:

ttl: 30

loop\_wait: 10

retry\_timeout: 10

maximum\_lag\_on\_failover: 1048576

postgresql:

use\_pg\_rewind: true

use\_slots: true

parameters:

initdb:

- encoding: UTF8

- data-checksums

pg\_hba:

- host replication replicator 127.0.0.1/32 md5

- host replication replicator 10.0.0.1/32 md5

- host replication replicator 10.0.0.2/32 md5

- host replication replicator 10.0.0.3/32 md5

- host all all 0.0.0.0/0 md5

users:

admin:

password: admin

options:

- createrole

- createdb

postgresql:

listen: 10.0.0.3:5432

connect\_address: 10.0.0.3:5432

data\_dir: /var/lib/pgsql/15/data

bin\_dir: /usr/pgsql-15/bin

pgpass: /tmp/pgpass

authentication:

replication:

username: replicator

password: replicator

superuser:

username: postgres

password: postgres

watchdog:

mode: required

device: /dev/watchdog

safety\_margin: 5

tags:

nofailover: false

noloadbalance: false

clonefrom: false

nosync: false

**CẤU HÌNH WATCHDOG**

**Watchdog** là các cơ chế phần mềm hoặc phần cứng sẽ khởi động lại toàn bộ hệ thống khi chúng không nhận được tín hiệu báo hiệu (heartbeat) trong một khoảng thời gian nhất định. Tính năng này bổ sung thêm một lớp bảo vệ an toàn phụ trợ trong trường hợp các cơ chế bảo vệ split-brain thông thường của Patroni bị lỗi.

**Patroni sẽ cố gắng kích hoạt watchdog trước khi chuyển PostgreSQL lên primary.** Hiện tại, watchdog chỉ được hỗ trợ bằng cách sử dụng giao diện thiết bị watchdog của Linux. Để biết thêm thông tin về watchdog, hãy xem tài liệu Patroni.

**Cấu hình Patroni mặc định sẽ cố gắng sử dụng /dev/watchdog trên Linux** nếu Patroni có thể truy cập. Đối với hầu hết các trường hợp sử dụng, sử dụng watchdog phần mềm được tích hợp sẵn trong nhân Linux là đủ an toàn.

* Watchdog là cơ chế đảm bảo hệ thống hoạt động ổn định bằng cách khởi động lại nếu mất liên lạc.
* Patroni kích hoạt watchdog trước khi chuyển đổi sang Primary để đảm bảo tính nhất quán dữ liệu.
* Watchdog sử dụng giao diện thiết bị watchdog của Linux.
* Cấu hình mặc định của Patroni sẽ thử sử dụng /dev/watchdog.
* Sử dụng watchdog phần mềm trong nhân Linux thường là đủ an toàn.

**TRÊN TẤT CẢ CÁC NODE:**

sudo vi /etc/watchdog.conf

## active dòng này lên, bỏ comment đi:

watchdog-device = /dev/watchdog

## thực hiện chuỗi lệnh bên dưới để active software watchdog

sudo mknod /dev/watchdog c 10 130

sudo modprobe softdog

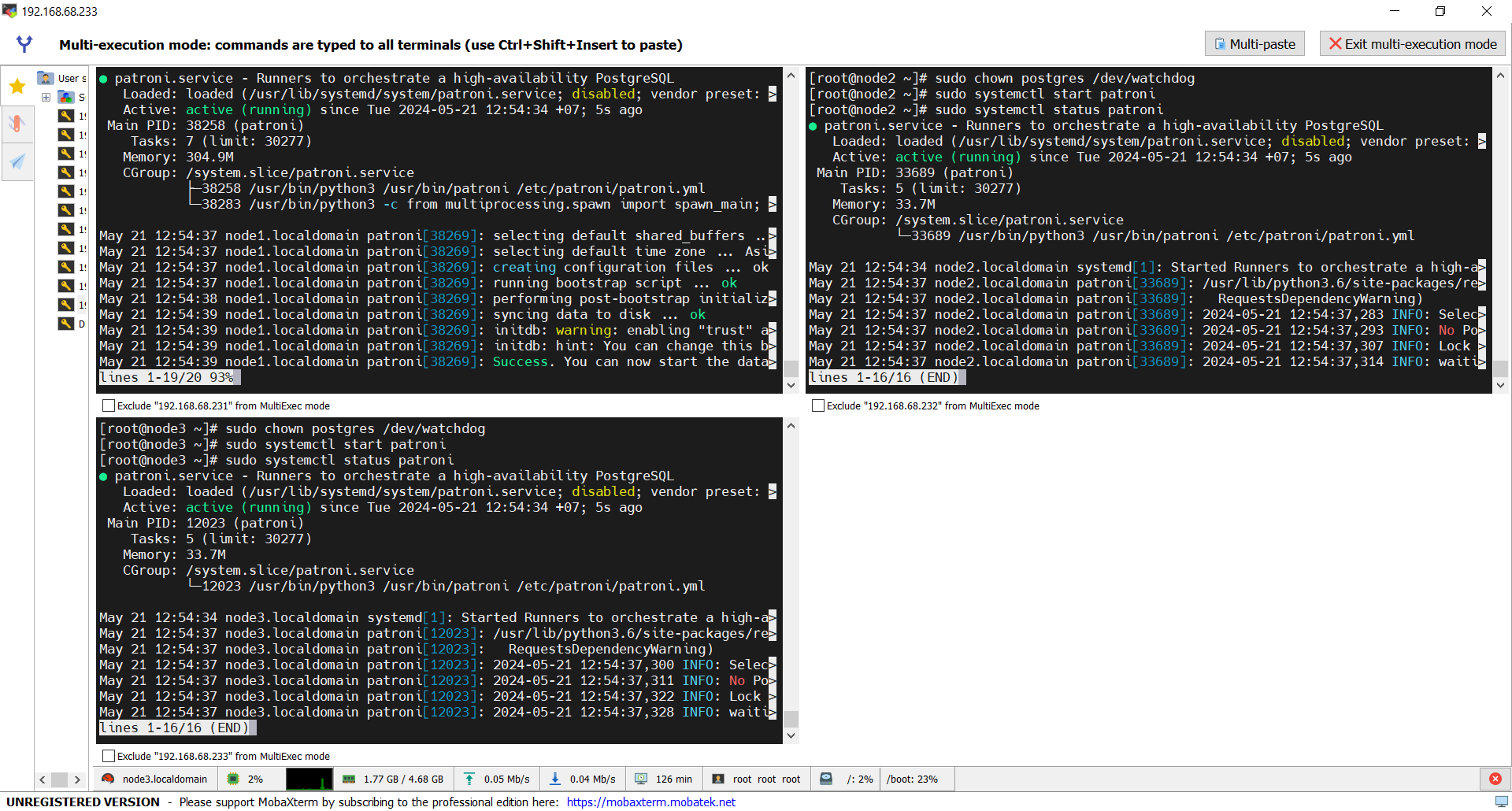
sudo chown postgres /dev/watchdog

**Bật Patroni Cluster**

**Node1, 2, 3:**

sudo systemctl start patroni

sudo systemctl status patroni



**Cài đặt PgBouncer:**

**Node1, 2, 3:**

sudo dnf install -y pgbouncer

Cấu hình PgBouncer Authentication

**Node1:**

su postgres

psql

#psql -h node1 -p 5432 -U postgres

## tạo user login và function PASSW: postgres

CREATE ROLE pgbouncer with LOGIN encrypted password 'postgres';

CREATE FUNCTION public.lookup (

INOUT p\_user name,

OUT p\_password text

) RETURNS record

LANGUAGE sql SECURITY DEFINER SET search\_path = pg\_catalog AS

$$SELECT usename, passwd FROM pg\_shadow WHERE usename = p\_user$$;

Copy pass đã được mã hoá của user pgbouncer và lưu lại:

select \* from pg\_shadow;

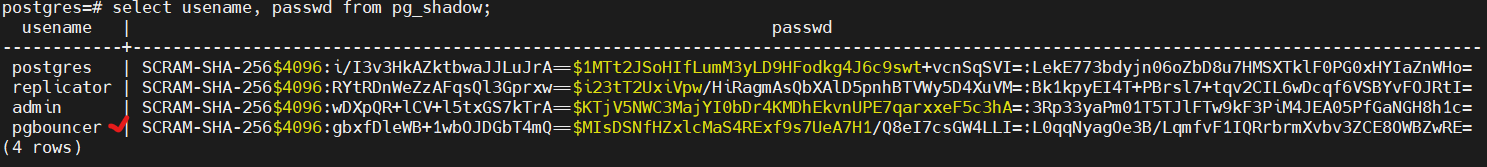
select usename, passwd from pg\_shadow;

## sửa thông tin file userlist (danh sách xác thực đăng nhập vào pgbouncer)

sudo vi /etc/pgbouncer/userlist.txt

## thêm vào file nội dung: trong đó phần mã hoá là copy ở bên trên

“pgbouncer” “SCRAM-SHA-256$4096:ZA4oeKbw7cEXDx5/9ka/Vg==$ywLS/Y/dcnfShhHr0W/TiAEg0mxxbRGT1aFgoifkoDs=:4eE3UvMTOwFKUvtOetc6g0Od7NEU/eE5wyCN3bJlQ7Q=”



Sửa file /etc/pgbouncer/pgbouncer.ini

sudo cp -p /etc/pgbouncer/pgbouncer.ini /etc/pgbouncer/pgbouncer.ini.orig

sudo vi /etc/pgbouncer/pgbouncer.ini

Tìm dòng có chữ [databases] và thêm vào bên dưới nội dung sau:

\* = host=10.0.0.1 port=5432 dbname=postgres

Tìm dòng có thông tin listen\_addr=localhost và đổi thành listen\_addr=\*

listen\_addr = \*

Tìm dòng auth\_file = /etc/pgbouncer/userlist.txt trong nhóm [pgbouncer] và thêm vào bên dưới dòng này nội dung sau:

auth\_user = pgbouncer

auth\_query = SELECT p\_user, p\_password FROM public.lookup($1)

**Node2:**

sudo vi /etc/pgbouncer/userlist.txt

## thêm vào nội dung sau: nội dung như node1 => thay thế bằng thông số của bạn

“pgbouncer” “SCRAM-SHA-256$4096:ZA4oeKbw7cEXDx5/9ka/Vg==$ywLS/Y/dcnfShhHr0W/TiAEg0mxxbRGT1aFgoifkoDs=:4eE3UvMTOwFKUvtOetc6g0Od7NEU/eE5wyCN3bJlQ7Q=”

sudo cp -p /etc/pgbouncer/pgbouncer.ini /etc/pgbouncer/pgbouncer.ini.orig

sudo vi /etc/pgbouncer/pgbouncer.ini

Tìm dòng có chữ [databases] và thêm vào bên dưới nội dung sau:

\* = host=10.0.0.2 port=5432 dbname=postgres

Tìm dòng có thông tin listen\_addr=localhost và đổi thành listen\_addr=\*

listen\_addr = \*

Tìm dòng auth\_file = /etc/pgbouncer/userlist.txt trong nhóm [pgbouncer] và thêm vào bên dưới dòng này nội dung sau:

auth\_user = pgbouncer

auth\_query = SELECT p\_user, p\_password FROM public.lookup($1)

**Node3:**

sudo vi /etc/pgbouncer/userlist.txt

## thêm vào nội dung sau:

“pgbouncer” “SCRAM-SHA-256$4096:ZA4oeKbw7cEXDx5/9ka/Vg==$ywLS/Y/dcnfShhHr0W/TiAEg0mxxbRGT1aFgoifkoDs=:4eE3UvMTOwFKUvtOetc6g0Od7NEU/eE5wyCN3bJlQ7Q=”

sudo cp -p /etc/pgbouncer/pgbouncer.ini /etc/pgbouncer/pgbouncer.ini.orig

sudo vi /etc/pgbouncer/pgbouncer.ini

Tìm dòng có chữ [databases] và thêm vào bên dưới nội dung sau:

\* = host=10.0.0.3 port=5432 dbname=postgres

Tìm dòng có thông tin listen\_addr=localhost và đổi thành listen\_addr=\*

listen\_addr = \*

Tìm dòng auth\_file = /etc/pgbouncer/userlist.txt trong nhóm [pgbouncer] và thêm vào bên dưới dòng này nội dung sau:

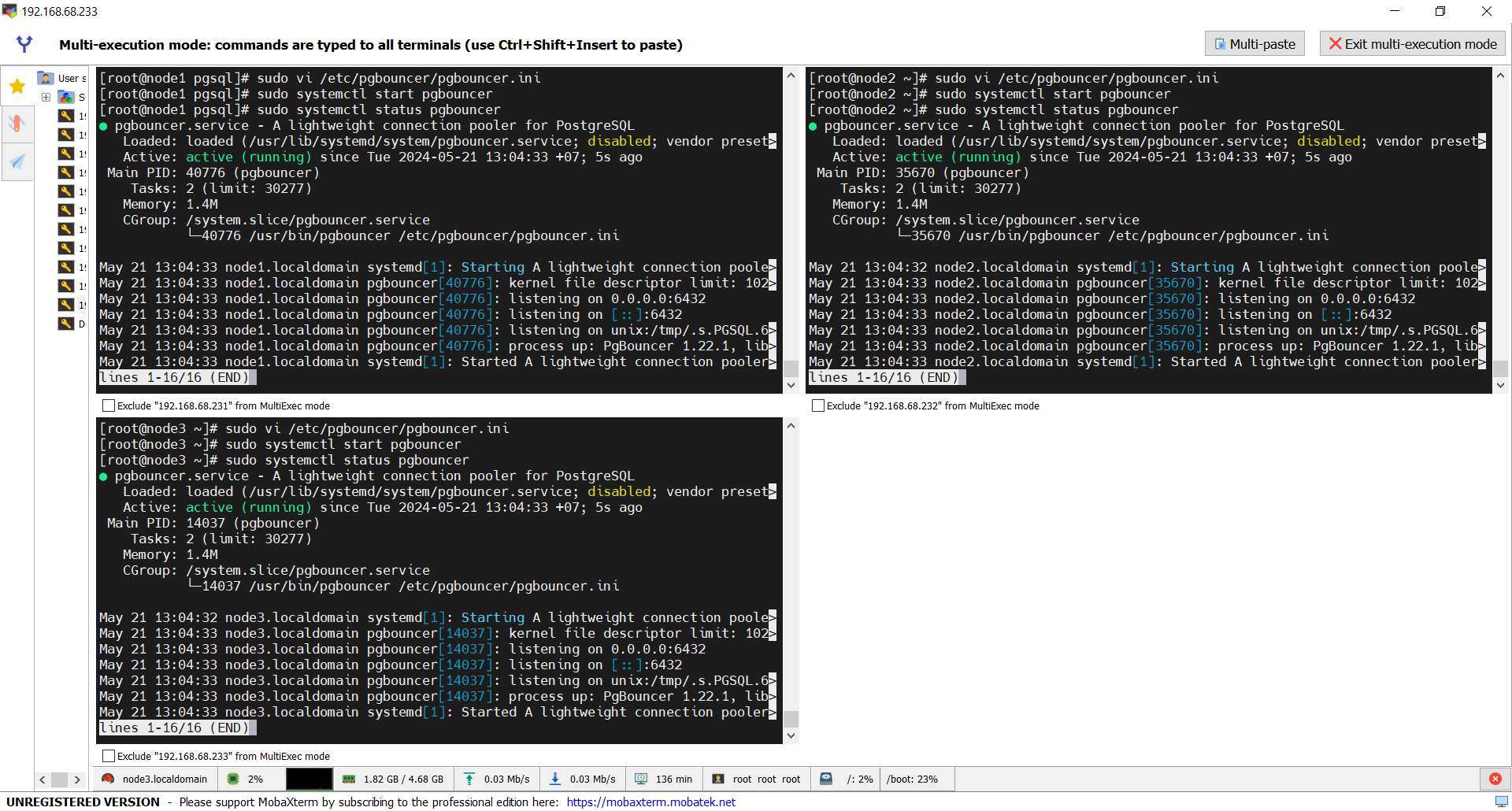
auth\_user = pgbouncer

auth\_query = SELECT p\_user, p\_password FROM public.lookup($1)

**Node1,2,3:**

sudo systemctl start pgbouncer

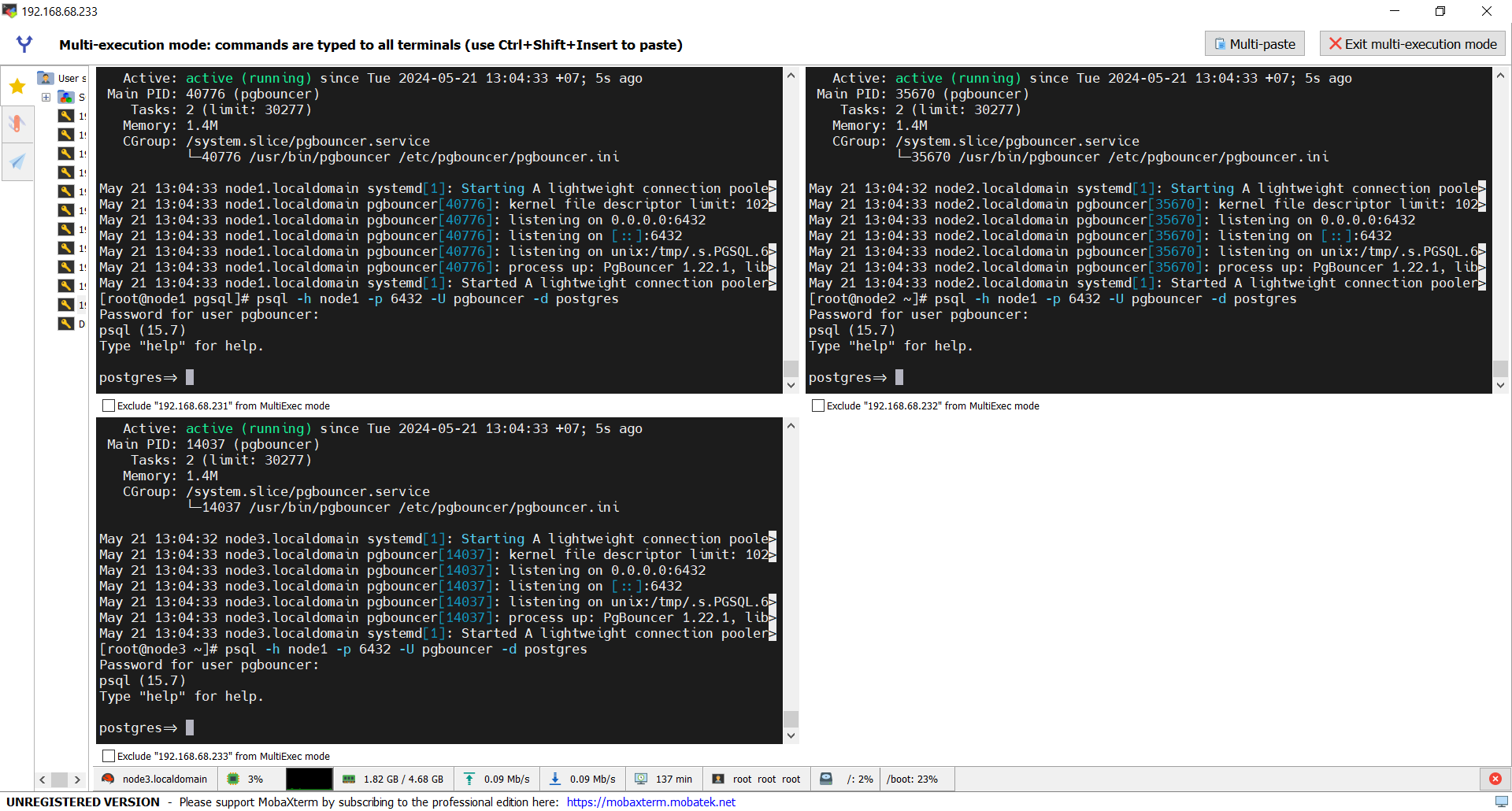
sudo systemctl status pgbouncer



**Test PgBouncer Authentication**

psql -h node1 -p 6432 -U pgbouncer -d postgres

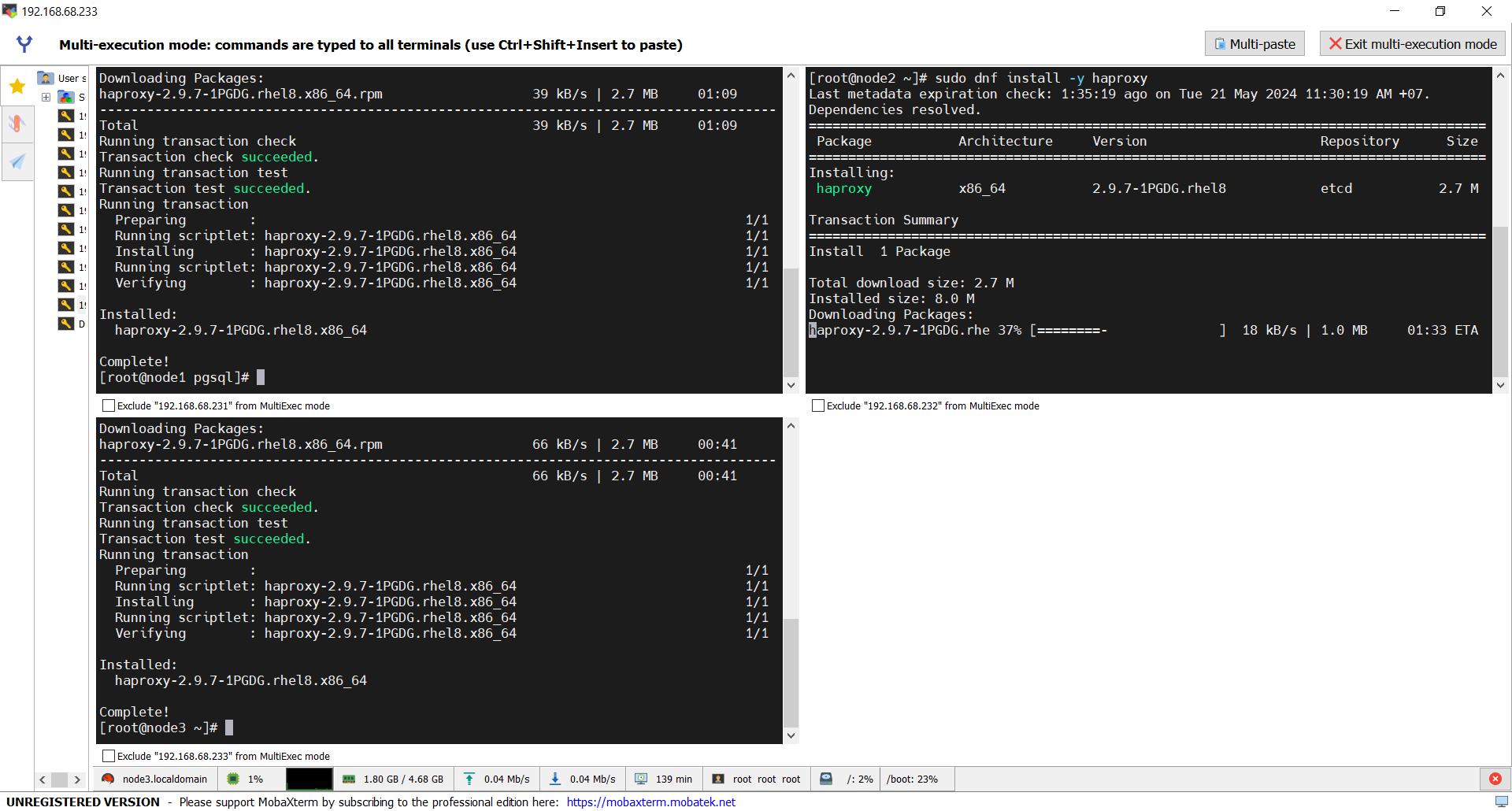
## kết quả:



**Cài đặt HAProxy:**

**Node1,2,3:**

sudo dnf install -y haproxy



**Cấu hình HAProxy:**

Node1,2,3:

sudo mv /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.orig

sudo vi /etc/haproxy/haproxy.cfg

##nội dung:

global

log 127.0.0.1 local2

chroot /var/lib/haproxy

pidfile /var/run/haproxy.pid

maxconn 1000

user haproxy

group haproxy

daemon

stats socket /var/lib/haproxy/stats

defaults

mode tcp

log global

option tcplog

retries 3

timeout queue 1m

timeout connect 10s

timeout client 1m

timeout server 1m

timeout check 10s

maxconn 900

listen stats

mode http

bind \*:7000

stats enable

stats uri /

listen primary

bind 10.0.0.100:5000

option httpchk OPTIONS /master

http-check expect status 200

default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions

server node1 10.0.0.1:6432 maxconn 100 check port 8008

server node2 10.0.0.2:6432 maxconn 100 check port 8008

server node3 10.0.0.3:6432 maxconn 100 check port 8008

listen standby

bind 10.0.0.100:5001

balance roundrobin

option httpchk OPTIONS /replica

http-check expect status 200

default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions

server node1 10.0.0.1:6432 maxconn 100 check port 8008

server node2 10.0.0.2:6432 maxconn 100 check port 8008

server node3 10.0.0.3:6432 maxconn 100 check port 8008

**primary** sử dụng port 5000 để reads/writes dữ liệu vào back-end database leader node.

**standby** sử dụng port 5001 để reads to back-end database replica nodes.

**Cài đặt Keepalived:**

**Node1,2,3:**

sudo dnf install -y keepalived

**Cấu hình Keepalived**

**Node1,2,3:**

sudo vi /etc/sysctl.conf

## nội dung

net.ipv4.ip\_nonlocal\_bind = 1

net.ipv4.ip\_forward = 1

**Apply thay đổi không cần reboot:**

sudo sysctl --system

sudo sysctl -p

**Node1:**

sudo mv /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf.orig

sudo vi /etc/keepalived/keepalived.conf

## noi dung

vrrp\_script chk\_haproxy {

script "pkill -0 haproxy"

interval 5

weight -4

fall 2

rise 1

}

vrrp\_script chk\_lb {

script "pkill -0 keepalived"

interval 1

weight 6

fall 2

rise 1

}

vrrp\_script chk\_servers {

script "echo 'GET /are-you-ok' | nc 127.0.0.1 7000 | grep -q '200 OK'"

interval 2

weight 2

fall 2

rise 2

}

vrrp\_instance vrrp\_1 {

interface enp0s3

state MASTER

virtual\_router\_id 51

priority 101

virtual\_ipaddress\_excluded {

10.0.0.100

}

track\_interface {

enp0s3 weight -2

}

track\_script {

chk\_haproxy

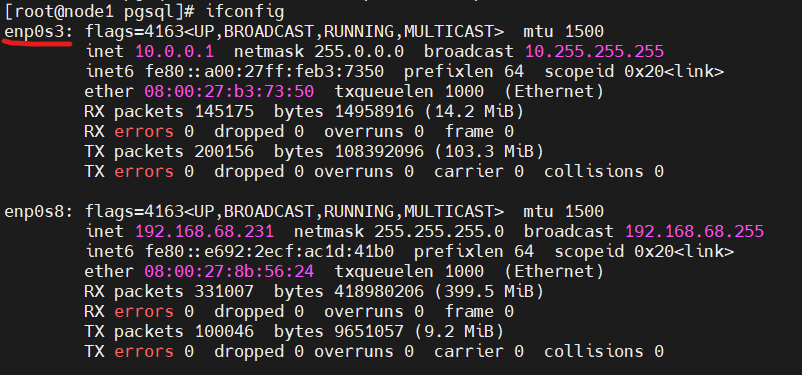
chk\_lb

}

}

Lưu ý: chỉnh sửa tên card mạng phù hợp với máy tính của bạn trong file bên trên

để kiểm tra tên card mạng thì chúng ta sử dụng lênh ifconfig.



**Node2:**

sudo mv /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf.orig

sudo vi /etc/keepalived/keepalived.conf

## noi dung

vrrp\_script chk\_haproxy {

script "pkill -0 haproxy"

interval 5

weight -4

fall 2

rise 1

}

vrrp\_script chk\_lb {

script "pkill -0 keepalived"

interval 1

weight 6

fall 2

rise 1

}

vrrp\_script chk\_servers {

script "echo 'GET /are-you-ok' | nc 127.0.0.1 7000 | grep -q '200 OK'"

interval 2

weight 2

fall 2

rise 2

}

vrrp\_instance vrrp\_1 {

interface enp0s3

state BACKUP

virtual\_router\_id 51

priority 100

virtual\_ipaddress\_excluded {

10.0.0.100

}

track\_interface {

enp0s3 weight -2

}

track\_script {

chk\_haproxy

chk\_lb

}

}

**Node3:**

sudo mv /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf.orig

sudo vi /etc/keepalived/keepalived.conf

## noi dung

vrrp\_script chk\_haproxy {

script "pkill -0 haproxy"

interval 5

weight -4

fall 2

rise 1

}

vrrp\_script chk\_lb {

script "pkill -0 keepalived"

interval 1

weight 6

fall 2

rise 1

}

vrrp\_script chk\_servers {

script "echo 'GET /are-you-ok' | nc 127.0.0.1 7000 | grep -q '200 OK'"

interval 2

weight 2

fall 2

rise 2

}

vrrp\_instance vrrp\_1 {

interface enp0s3

state BACKUP

virtual\_router\_id 51

priority 99

virtual\_ipaddress\_excluded {

10.0.0.100

}

track\_interface {

enp0s3 weight -2

}

track\_script {

chk\_haproxy

chk\_lb

}

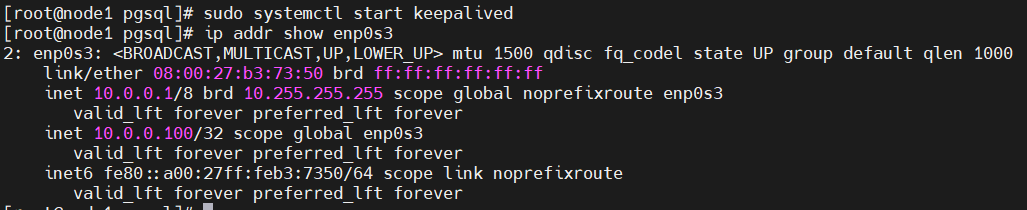
}

**Node1,2,3:**

sudo systemctl start keepalived

ip addr show enp0s3

##kết quả:

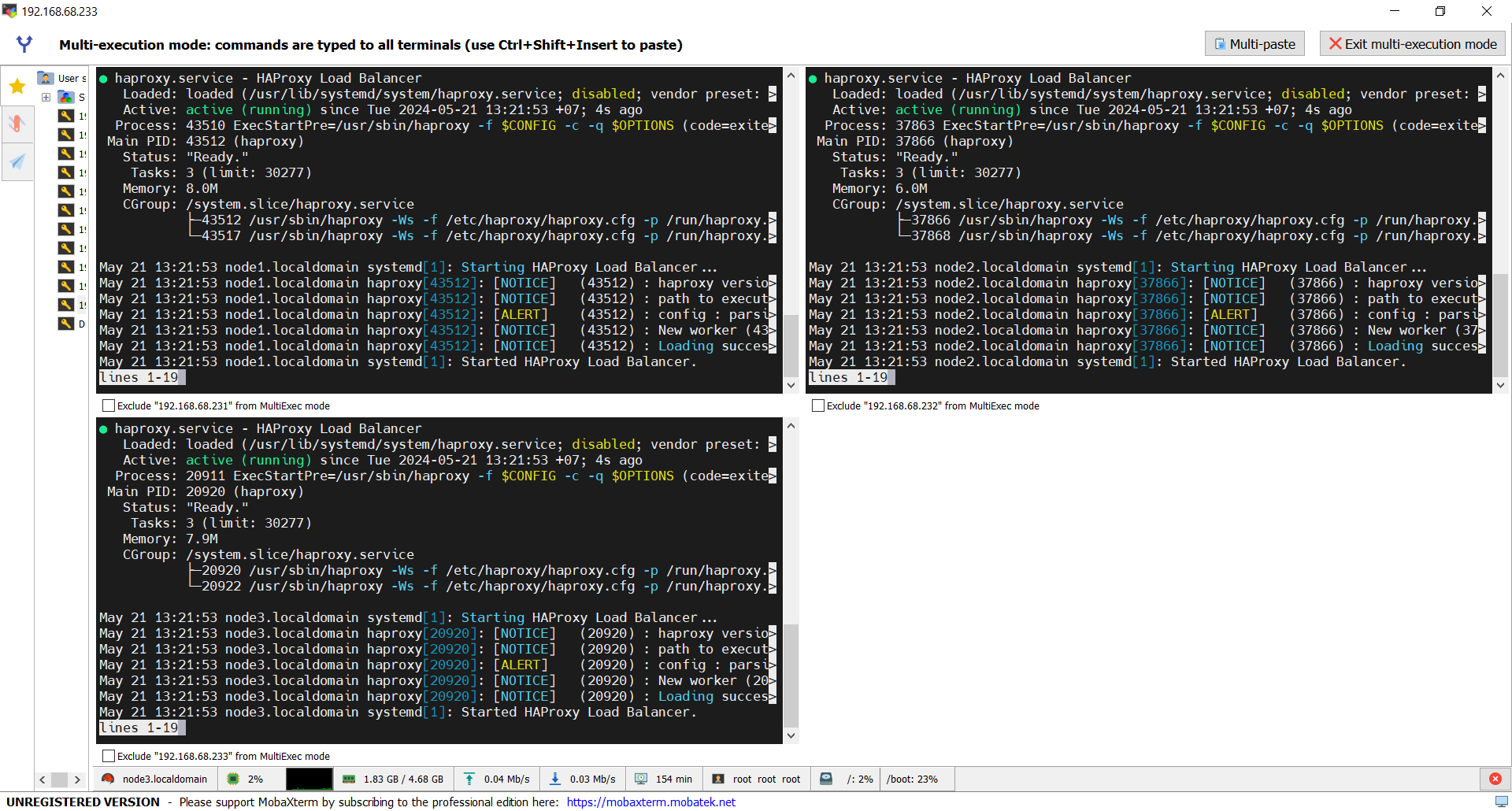


Với keepalived configuration, nếu MASTER haproxy down, shared IP sẽ tự động chuyển sang node BACKUP haproxy node, kết nối được active bình thường.

**Node1,2,3:**

sudo systemctl start haproxy

sudo systemctl status haproxy



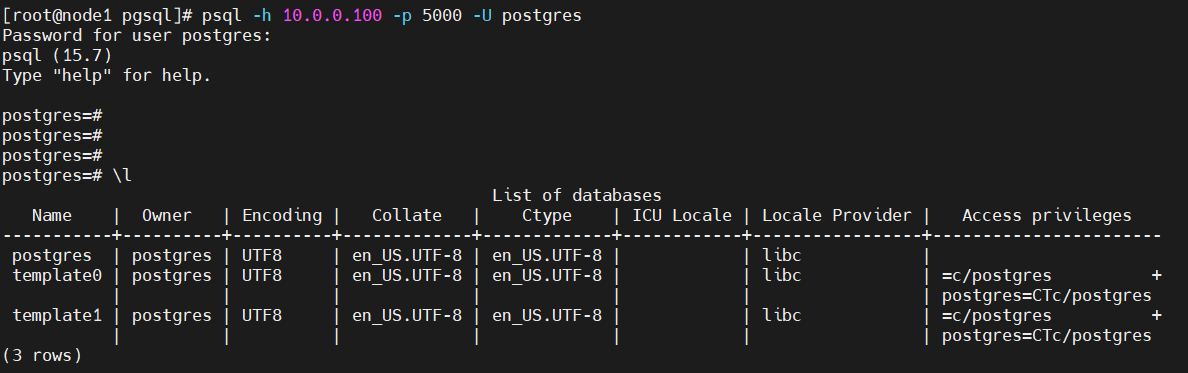
**Test Patroni Cluster**

**Node1:**

psql -h 10.0.0.100 -p 5000 -U postgres

## mật khẩu đã tạo ở bên trên cho pgbouncer: postgres

## kết quả:

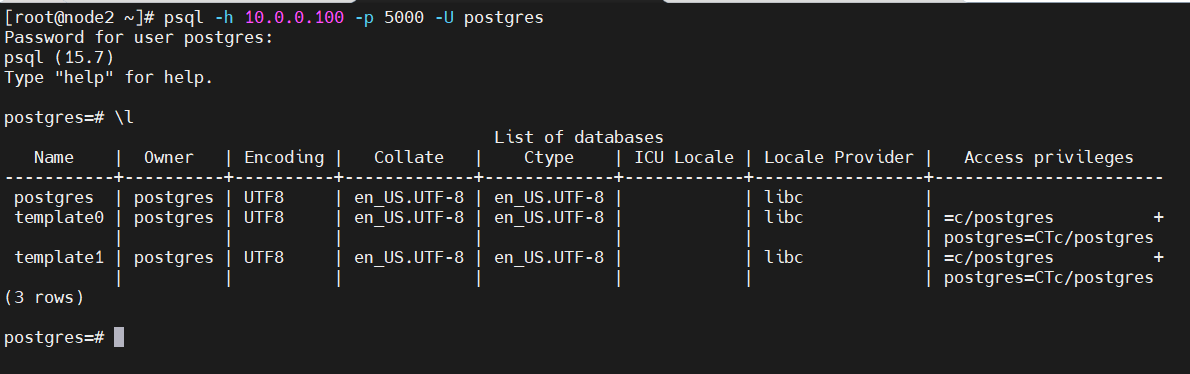


**Node2, 3:**

psql -h 10.0.0.100 -p 5000 -U postgres

psql -h 10.0.0.100 -p 5001 -U postgres -t -c "select inet\_server\_addr()"

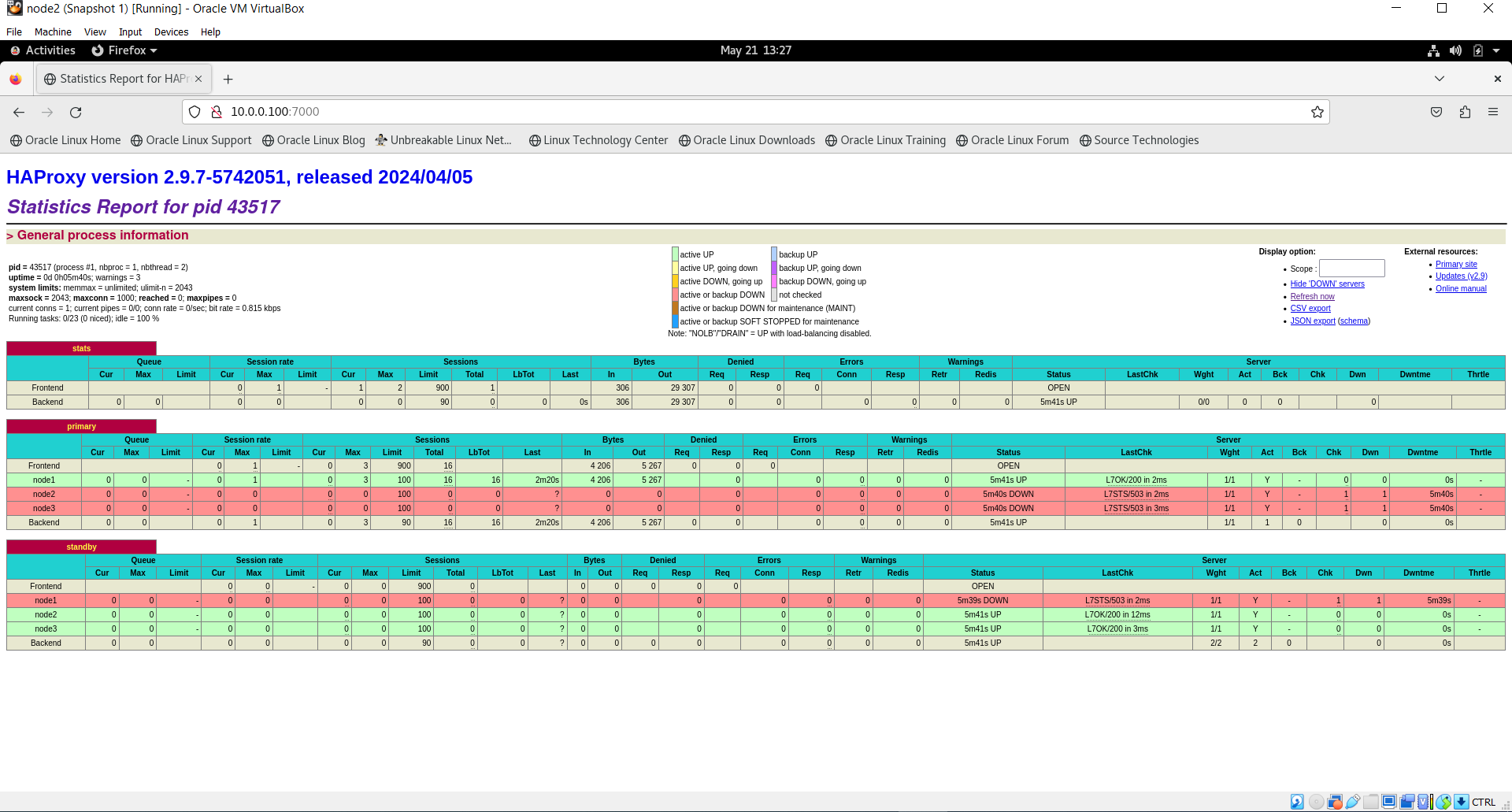
## kết quả:



**Truy cập HAProxy dashboard trên web:**

<http://10.0.0.100:7000/>

## kết quả:



Node1 đang là primary Read/Write

Node2,3 đang là standby Read only

**Test Postgres Database Replication:**

**Node1:**

psql -h 10.0.0.100 -p 5000 -U postgres

create database testdb;

create user testuser with encrypted password 'postgres';

grant all privileges on database testdb to testuser;