Enable Postgres SSL on pg_auto_failover

The host list:

- monitor: pg-auto-failover-monitor (192.168.6.165)
- node1: pg-auto-failover-node1 (192.168.6.166)
- node2: pg-auto-failover-node2 (192.168.6.167)

Step#1: Create the certs

- - Create the ROOT cert, the CN must include the user name "postgres"

```
mkdir -pv ~/cert; cd ~/cert

openssl req -new -nodes -text -out root.csr \
  -keyout root.key -subj "/CN=pg-auto-failover-monitor/CN=postgres"

openssl x509 -req -in root.csr -text -days 3650 \
  -extfile /etc/pki/tls/openssl.cnf -extensions v3_ca \
  -signkey root.key -out root.crt
```

- - Create the server cert for the monitor, this cert will sign by the root cert, the CN must be the same one as when we used to create the monitor (in --hostname)

```
openssl req -new -nodes -text -out server.csr \
  -keyout server.key -subj "/CN=pg-auto-failover-monitor"

openssl x509 -req -in server.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out server.crt
```

- - create client cert, note: in the CN, we use "postgres" as the user which will connect to DB

later we will set pg_hba.conf and pg_ident.conf, which will map postgres to

autoctl_node

(for connect to monitor) or

pgautofailover_replicator

(for replication between nodes)

```
openssl req -new -nodes -text -out client.csr \
  -keyout client.key -subj "/CN=postgres"

openssl x509 -req -in client.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out client.crt

chmod 600 *
```

Then we will create certs for the data nodes

```
cd ~/cert
mkdir node1
mkdir node2
cp root.* node1/
cp root.* node2/

#### Create certs for node 1 ####
cd node1

## server cert
openssl req -new -nodes -text -out server.csr \
  -keyout server.key -subj "/CN=pg-auto-failover-node1/CN=node1"

openssl x509 -req -in server.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out server.crt

### client cert
openssl req -new -nodes -text -out client.csr \
  -keyout client.key -subj "/CN=postgres"

openssl x509 -req -in client.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out client.crt

chmod 600 *

### send to node1
cd ..; ssh node1 "rm -rfv ~/cert"; scp -r ~/cert/node1/ node1:~/cert


#### Create certs for node 2 ####
```

```
cd node2

## server certs
openssl req -new -nodes -text -out server.csr \
  -keyout server.key -subj "/CN=pg-auto-failover-node2/CN=node2"

openssl x509 -req -in server.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out server.crt

## client certs
openssl req -new -nodes -text -out client.csr \
  -keyout client.key -subj "/CN=postgres"

openssl x509 -req -in client.csr -text -days 365 \
  -CA root.crt -CAkey root.key -CAcreateserial \
  -out client.crt

chmod 600 *

## send to node2
cd ..; ssh node2 "rm -rfv ~/cert"; scp -r ~/cert/node2/ node2:~/cert
```

- - At this point, we have created certs for all nodes

Step#2: Set up the monitor

- - Create the monitor by using the certs

```
/opt/vmware/postgres/13/bin/pg_autoctl create monitor \
--ssl-ca-file /home/postgres/cert/root.crt \
--server-cert /home/postgres/cert/server.crt \
--server-key /home/postgres/cert/server.key \
--ssl-mode verify-full \
--pgdata $PGDATA \
--hostname pg-auto-failover-monitor \
--auth trust \
--pgctl /opt/vmware/postgres/13/bin/pg_ctl
```

- - Once done, update the Postgres settings

(we will set user mapping to let

postgres

user map to

autoctl_node

, this is for letting the data node connect to monitor with certs)

```
### in pg_hba.conf ###
# allow certificate based authentication to the monitor
hostssl pg_auto_failover autoctl_node 192.168.6.165/32 cert map=pgautofailover
hostssl pg_auto_failover autoctl_node 192.168.6.166/32 cert map=pgautofailover
hostssl pg_auto_failover autoctl_node 192.168.6.167/32 cert map=pgautofailover

### in the pg_ident.conf ###
# pg_autoctl runs as postgres and connects to the monitor autoctl_node user
pgautofailover   postgres               autoctl_node
```

- - restart the monitor

```
sudo systemctl restart pgautofailover
```

Step#3: Setup data node

Setup data node1:

- - copy the client certs, when data node connect to monitor it will use those client certs

```
mkdir -pv  ~/.postgresql
cp -rp ~/cert/client.key ~/.postgresql/postgresql.key
cp -rp ~/cert/client.crt ~/.postgresql/postgresql.crt
cp -rp ~/cert/root.crt ~/.postgresql/root.crt
```

- - create postgres on node1

```
pg_autoctl create postgres \
--pgdata /data/postgresql/node1 \
--auth trust \
--ssl-ca-file /home/postgres/cert/root.crt \
--server-cert /home/postgres/cert/server.crt \
--server-key /home/postgres/cert/server.key \
--username postgres \
--hostname pg-auto-failover-node1 \
--pgctl /opt/vmware/postgres/13/bin/pg_ctl \
--monitor 'postgres://autoctl_node@pg-auto-failover-monitor:5432/pg_auto_failover?sslmode=
verify-full'
```

add the below line into the config file of node1

```
#### add below into pg_hba.conf ####
# allow streaming replication for pg_auto_failover nodes
hostssl replication pgautofailover_replicator 192.168.6.166/32 cert map=pgautofailover
hostssl replication pgautofailover_replicator 192.168.6.167/32 cert map=pgautofailover

hostssl "postgres" pgautofailover_replicator pg-auto-failover-node2 cert map=pgautofailove
r
hostssl replication "pgautofailover_replicator" pg-auto-failover-node2 trust # Auto-genera
ted by pg_auto_failover
hostssl "postgres" "pgautofailover_replicator" pg-auto-failover-node2 trust # Auto-generat
ed by pg_auto_failover

### add below into pg_ident.conf ###
# pgautofailover_replicator runs as postgres and do replication to other nodes
pgautofailover  postgres               pgautofailover_replicator
```

restart the node1

```
sudo systemctl restart pgautofailover
```

Setup data node2

- - copy the client certs

```
mkdir -pv  ~/.postgresql
cp -rp ~/cert/client.key ~/.postgresql/postgresql.key
cp -rp ~/cert/client.crt ~/.postgresql/postgresql.crt
cp -rp ~/cert/root.crt ~/.postgresql/root.crt
```

- - create the data node

```
pg_autoctl create postgres \
--pgdata /data/postgresql/node2 \
--auth trust \
--ssl-ca-file /home/postgres/cert/root.crt \
--server-cert /home/postgres/cert/server.crt \
--server-key /home/postgres/cert/server.key \
--username postgres \
--hostname pg-auto-failover-node2 \
--pgctl /opt/vmware/postgres/13/bin/pg_ctl \
--monitor 'postgres://autoctl_node@pg-auto-failover-monitor:5432/pg_auto_failover?sslmode=
verify-full'
```

check if we have the correct settings, those settings should be replicated from primary node1

```
## pg_hba.conf ##
hostssl replication pgautofailover_replicator 192.168.6.166/32 cert map=pgautofailover
hostssl replication pgautofailover_replicator 192.168.6.167/32 cert map=pgautofailover

hostssl "postgres" pgautofailover_replicator pg-auto-failover-node2 cert map=pgautofailove
r
hostssl replication "pgautofailover_replicator" pg-auto-failover-node2 trust # Auto-genera
ted by pg_auto_failover
hostssl "postgres" "pgautofailover_replicator" pg-auto-failover-node2 trust # Auto-generat
ed by pg_auto_failover

### pg_ident.conf ###
# pgautofailover_replicator runs as postgres and do replication to other nodes
pgautofailover  postgres               pgautofailover_replicator
```

restart the node2

```
sudo systemctl restart pgautofailover
```

At this point, we are done with all steps, go back to the monitor node, and verify the cluster is working. we can also do some test to failover to make sure everything is working

```
$ pg_autoctl show state
  Name |  Node |                  Host:Port |       TLI: LSN |   Connection |      Report
ed State |      Assigned State
-------+-------+----------------------------+----------------+--------------+------------
---------+--------------------
node_1 |     1 | pg-auto-failover-node1:5432 |   1: 0/3000110 |   read-write |
primary |             primary
node_2 |     2 | pg-auto-failover-node2:5432 |   1: 0/3000110 |    read-only |           s
econdary |           secondary

$ pg_autoctl perform failover
$ pg_autoctl show state
  Name |  Node |                  Host:Port |       TLI: LSN |   Connection |      Report
ed State |      Assigned State
-------+-------+----------------------------+----------------+--------------+------------
---------+--------------------
node_1 |     1 | pg-auto-failover-node1:5432 |   2: 0/3000400 |    read-only |           s
```

```
econdary |           secondary
node_2 |     2 | pg-auto-failover-node2:5432 |   2: 0/3000400 |   read-write |
primary |             primary
```