BAB V
KESIMPULAN

5.1. Kesimpulan

Berdasarkan dengan pembahasan-pembahasan pada bab I hingga bab IV mengenai skripsi yang berjudul "Optimalisasi Kinerja Server Database PostgreSQL Melalui Cluster Patroni" penulis dapat menarik beberapa kesimpulan sebagai berikut:

1. Dalam menjalankan *Cluster* Patroni dibutuhkan beberapa *depedency package* yang harus *diinstall* yaitu etcd, HAProxy dan Keepalived.
2. Fitur *autofailover* yang disediakan oleh *Cluster* Patroni sudah mendukung untuk terciptanya *enviroment database* PostgreSQL yang memiliki *high availability*, karena fitur *autofailover* tersebut dapat menggantikan server *leader* secara otomatis jika terjadi kegagalan.
3. Dari hasil pengujian fungsional baik pengujian melalui aplikasi web POS maupun proses *dump* & *restore* kemampuan *high availability* yang dimiliki oleh *Cluster* Patroni dapat berjalan ketika *failure tolerance* dari *Cluster* etcd tidak melampaui batas yang sudah ditentukan, dengan adanya kemampuan *high avaiability* yang dimiliki oleh *Cluster* Patroni kinerja dari server *database* lebih optimal dan minim akan *downtime* atau kegagalan yang terjadi.

5.2. Saran

Berdasarkan beberapa kesimpulan yang tersebut diatas, penulis memberikan beberapa saran guna proses pengembangan yang lebih lanjut dalam implementasi *Cluster Patroni* sebagai berikut:

1. Dengan adanya *failure toleranc*e yang dibutuhkan oleh *service* etcd, akan lebih aman ketika jumlah *member* dari *Cluster* etcd semakin banyak. Jika *Cluster* Patroni yang dibangun hanya memiliki 2 member saja, maka untuk *service* etcd akan lebih baik jika diinstall disalah satu *member* Patroni saja atau etcd berjalan secara *standalone*.
2. Pada konfigurasi PostgreSQL dapat dilakukan *tuning configuration* sesuai dengan kebutuhan, *tuning* yang dimaksud diantaranya adalah penyesuaian maksimum koneksi, koneksi pool, *memory buffer* dan lain sebagainya yang terdapat pada file *postgresql.conf* guna menyesuaikan antara spesifikasi server dan kebutuhan dari user.
3. Ketika proses *promote leader* dari Cluster Patroni berjalan, membutuhkan waktu sekitar 10 – 20 detik untuk perpindahan server *leader*. Proses ini dapat memungkinkan aplikasi yang mengakses ke dalam database mengalami *connection timeout*. Alangkah baiknya dapat ditambahkan *variable* atau *handling value* maksimal *connection timeout* dari sisi aplikasi tersebut untuk memitigasi *connection timeout* yang kemungkinan terjadi ketika proses perpindahan server *leader*.

4. Ketika terjadi perpindahan server *leader* pada *Cluster* Patroni tidak terdapat notifikasi sehingga harus melakukan pengecekan secara manual. Untuk notifikasi perpindahan server *leader* dapat diimplementasikan melalui *bashscript* dan bisa diintegrasikan dengan Telegram atau sebagainya.
5. Untuk penelitian yang dilakukan oleh penulis mengenai implementasi *Cluster* Patroni sejauh ini baru dilakukan pada Sistem Operasi yang bersifat *Open Source*, Bagi para pembaca dapat melakukan ujicoba implementasi pada Sistem Operasi *Close Source* seperti Windows Server atau semacamnya.