

Univerzitet u Nišu

Elektronski fakultet Niš

Seminarski rad

**High availability primeru Redis baze podataka**

Sistemi za upravljanje bazama podataka 2019/20

Mentor: Student:  
Aleksandar Stanimirović Todor Majstorović 1088

# Uvod

Često se može desiti pad performansi servera, poput naglog zagušenja ili nestanka struje. Da bi se takvi slučajevi sprečili, potrebno je osposobiti sistem da se izbori sa takvim problemima. Rešenje je korišćenje High Availability (HA) arhitekture, gde se servisi implementiraju tako da osiguraju optimalan rad servera. Iako ne postoje jedinstvena rešenja, postoje preporuke koje se trebaju ispoštovati.

## 1. Koristiti više servera

Ukoliko se koristi samo jedan server, pad tog servera dovodi do pada cele aplikacije dok se taj server ne restartuje.

Očigledno rešenje je deployovanje vaše aplikacije na više servera, vodeći računa o pravilnoj raspodeli zahteva na svaki od tih servera. Moguće je čak i rasporediti delove servera, poput dela koji procesira fajlove ili se bavi obradom mailova.

## 2. Skaliranje baza podataka

Baze podataka su najpopularniji način za čuvanje podataka. Podjednako su važne kao i sam server. Baze se pokreću na odvojenim serverima i moguće je da taj server otkaže, što može dovesti do gubljenja korisničkih podataka.

Redundansa je proces kreiranja sistema sa više nivoa dostupnosti koji može da detektuje otkaz i da se automatski oporavi. Ovo se postiže održavanjem slave baza, koje se ažuriraju zajedno sa glavnom bazom. Još jedan bitan koncept je shardovanje. Shard je horizontalna particija baze, gde se redovi tabela dele na više servera.

# Redis

Redis je open-source in-memory baza podataka koja je dizajnirana da bude brza i jednostavna. Omogućava real-time komunikaciju, većina upita se izvršavaju kraće od milisekunde što dopušta veliki broj konkurentnih upita. Kako bi osigurao visoku dostupnost, oporavak od grešaka i replikaciju, koriste se Redis cluster i Redis Sentinel.

Instanca Redisa predstavlja proces koji se ponaša kao in-memory skladište. Svi upiti za čitanje i upis idu ka tom procesu. Kako bi se zaštitili od otkaza, Redis može da replicira podatke, kao i da šalje promene većem broju instanca ili replika.

Redis koristi unidirekcionu asinhronu replikaciju od mastera ka slave-ovima. Jedan master može imati veći broj replika, a replike mogu imati svoje replike. Kako bi se osiguralo što kraće kašnjenje, replikacije je neblokirajuća i izvršava se nakon što se izvrši na masteru. Moguće je da se izgube podaci ako se otkaz desi izmedju izvršenja operacije i izvršenja replikacije. Jedno rešenje je da se čeka dok se ne napravi odredjen broj replika.

Redis replike mogu biti dodavane i konfigurisane u runtime, a podržana je i parcijalna replikacija. Replike mogu da se podese da služe kao hot-standby, da opslužuju samo upite za čitanje, a mogu čak i da dozvole upise.

Pros:

* Lako i jednostavno za podešavanje
* Može biti automatizovano
* Nastavlja da radi dok je jedna instanca mastera dostupna

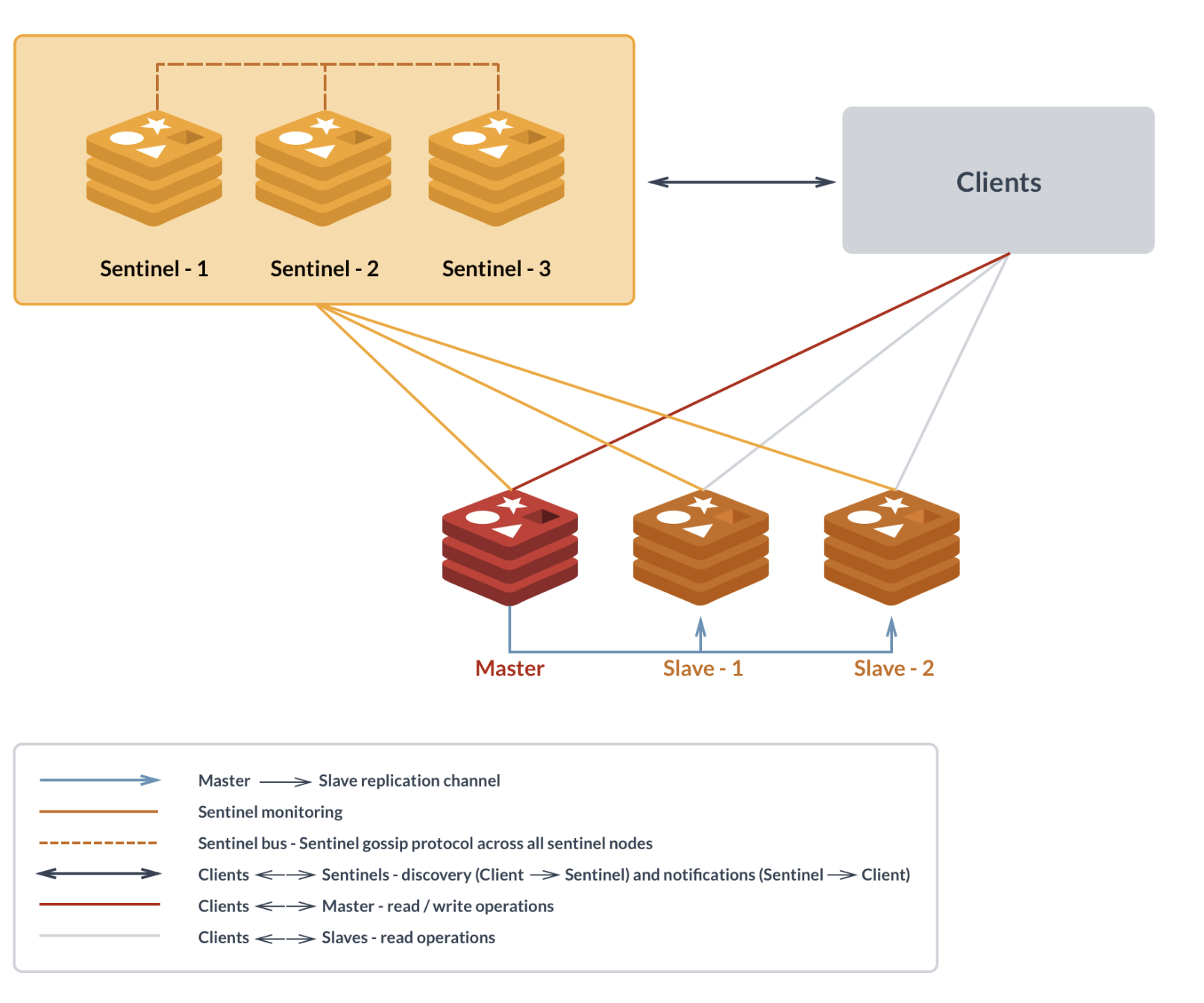
Cons:

* Upisi moraju da idu na master
* Slave može da opslužuje čitanje, ali mogu se dobiti nevalidni podaci
* Ne sharduje podatke
* Ako otkaže master, ručno se mora unaprediti novi

Problem kod Redis replika je što u slučaju otkaza mastera mora ručno da se unapredi slave u novi master. Ovaj problem se rešava korišćenjem Redis Sentinela.

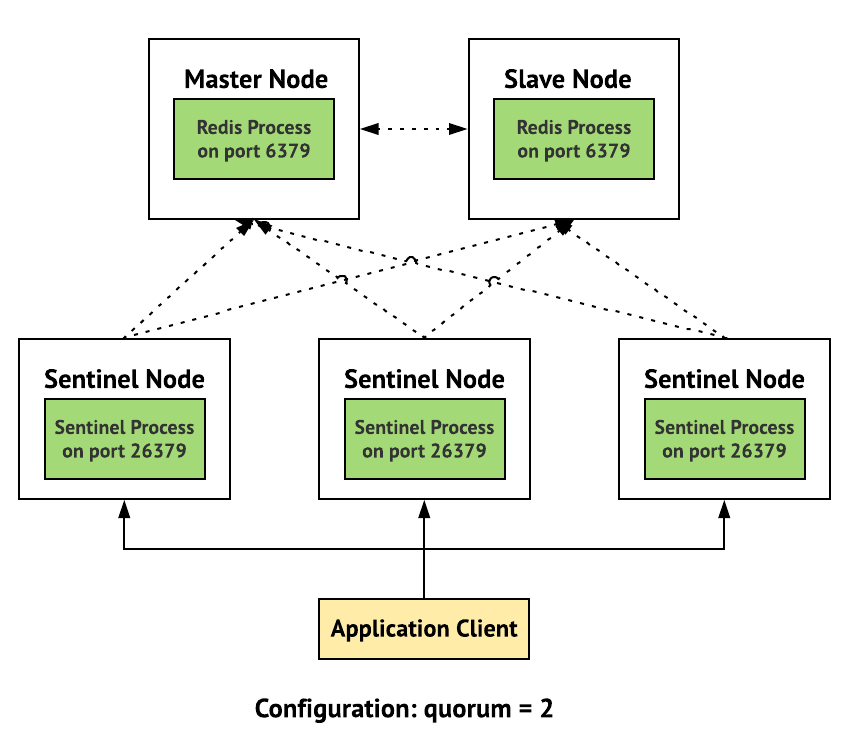
# Redis Sentinel

Kada se koristi Redis sa jednom instancom, gde jedan Redis server opslužuje celu bazu, Redis Sentinel se bavi dostupnošću. Sentinel je komponenta Redis servera koja se mora deployovati pored servera i replika. Sentinel se zasniva na kvorumu kako bi otkrio otkaz, pa zahteva više instanci koje će se nadgledati. Kada se većina Sentinela složi da je otkaz nastupio, jedna od replika postaje master. Redis Sentinel pruža i interfejs za otkrivanje servisa koje klijent koristi da se poveže sa glavnom Redis instancom.



Kada Sentinel detektuje otkaz, jedna od replika postaje master. Svi ostali slave-ovi se automatski konfigurišu da koriste novi master. Kada klijent postavi upit na Sentinel, Sentinel mu šalje adresu mastera.

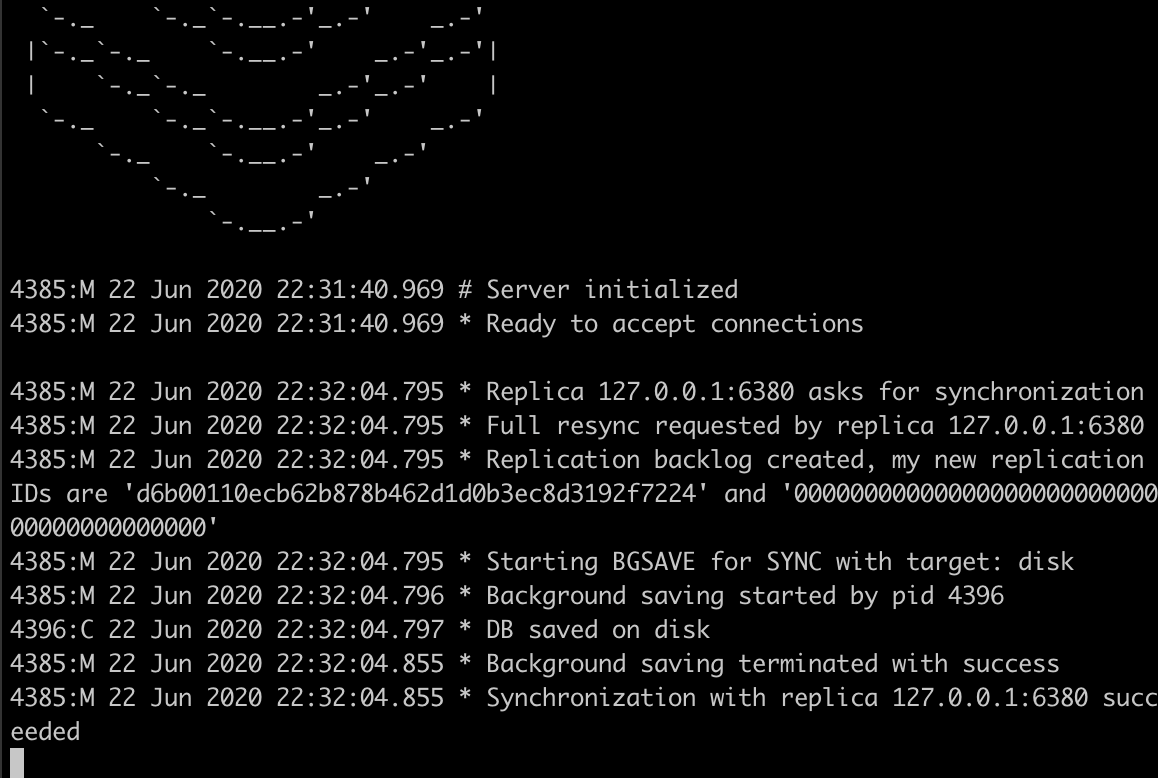
Kako Sentinel izvršava automatski oporavak:



Quorum vrednost predstavlja broj sentinela koji treba da potvrde otkaz. Medjutim to nije dovoljno, jedan od sentinela mora da se proglasi vodjom i da se autorizuje.

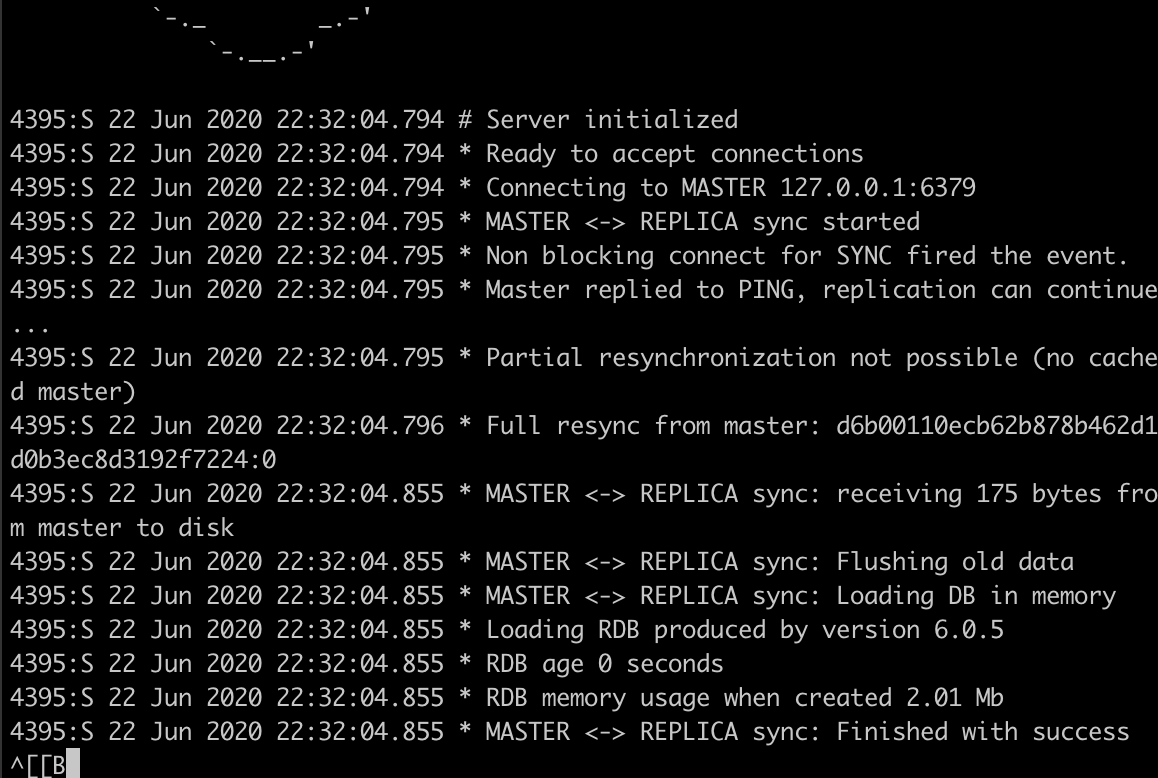
# Podešavanje replika i Sentinela

Pokrenuti master na portu 6379



Pokrenuti repliku komandom :

redis-server --port 6380 --slaveof 127.0.0.1 6379



#Master 1 sentinel 1

sentinel monitor mymaster 127.0.0.1 6379 2

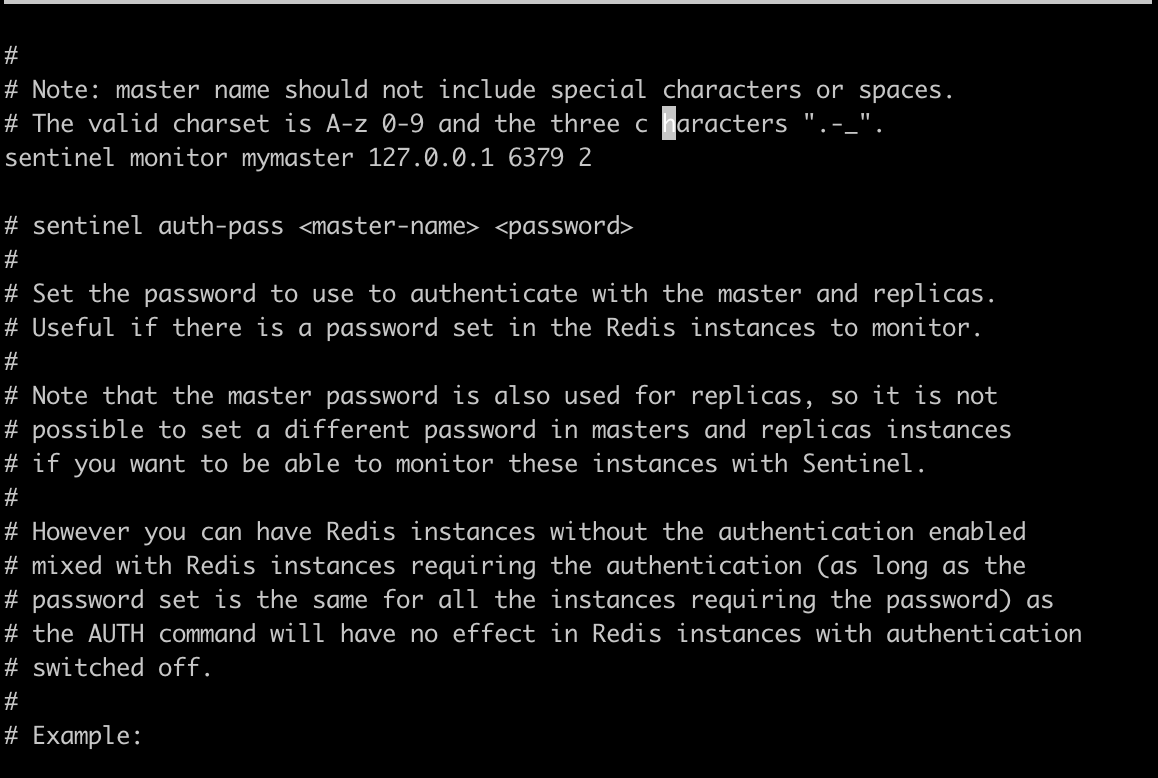
#Replika 1 sentinel 2

sentinel monitor mymaster 127.0.0.1 6380 2

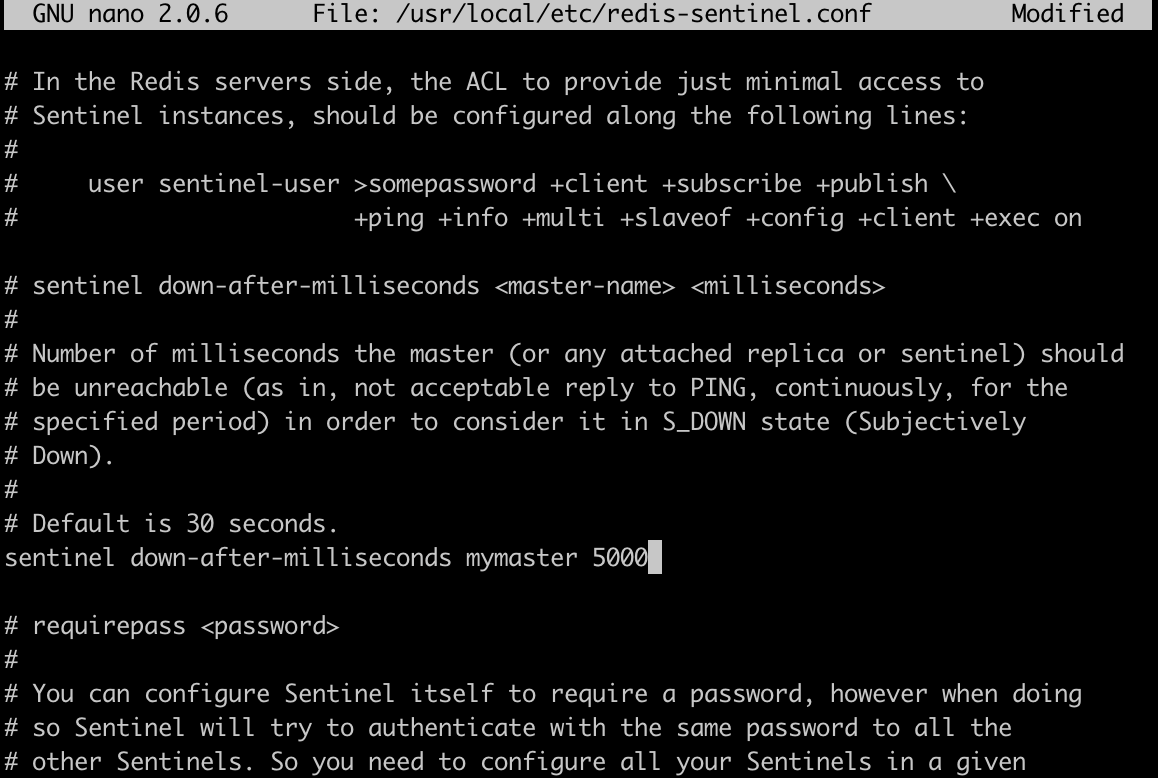
#Replika 1 sentinel 3

sentinel monitor mymaster 127.0.0.1 6380 2

Podesite port sentinela i adresu koju će sentinel da nadgleda



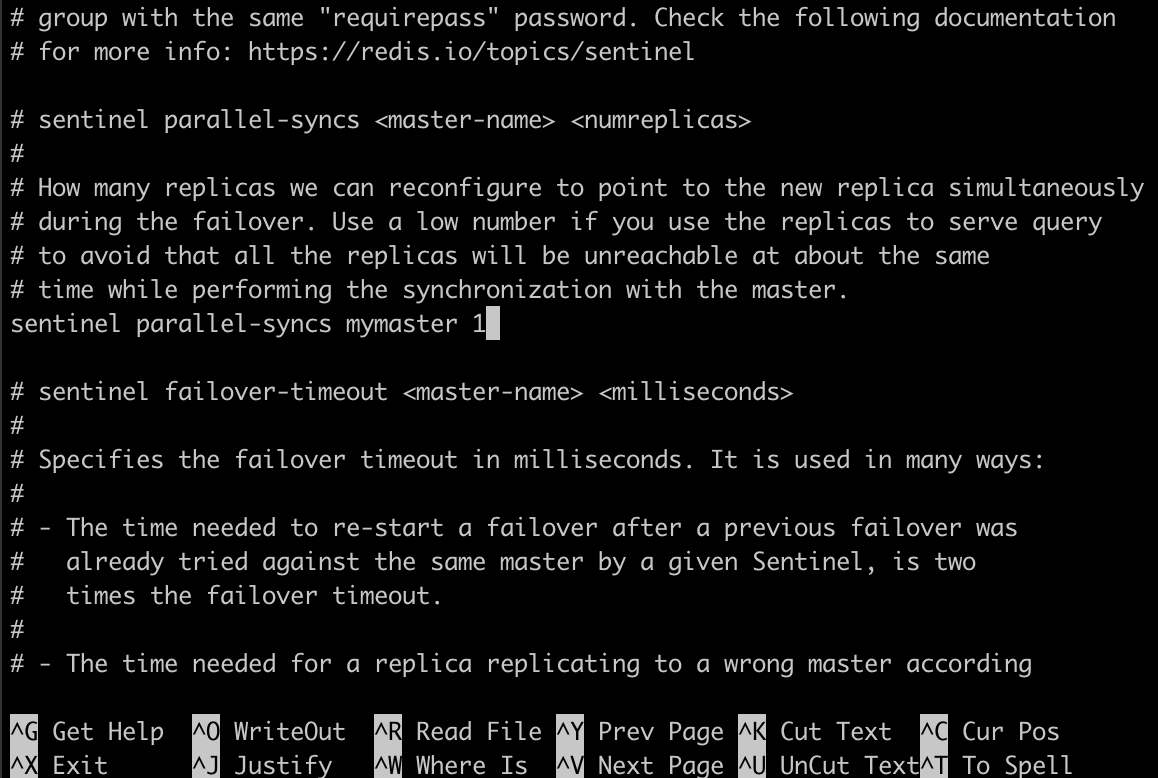
Broj 2 predstavlja podešavanje kvoruma



Podesite vreme koliko treba da prodje da se promeni stanje mastera nakon što padne.



Podesite broj replika koje se trebaju rekonfigurisati da koriste novi master nakon failovera, u našem slučaju pošto imamo 2 replike, 1 je potrebno podesiti



Restartujte Sentinel servis na svim čvorovima.

systemctl restart redis-sentinel

Restartujte Sentinel servis na svim čvorovima i otvorite portove kako bi Sentinel mogao da prima poruke na svom portu

firewall-cmd --zone=public --permanent --add-port=26379/tcp

firewall-cmd --reload

Sve replike ce biti automatski otkrivene nakon pokretanja sve tri instance sentinela.



Sentinel 1



Sentinel 2



Sentinel 3

127.0.0.1:5000> SENTINEL get-master-addr-by-name mymaster

1) "127.0.0.1"

2) "6379"

Testiranje Failovera

redis-cli -p 6379 DEBUG sleep 30

127.0.0.1:5000> SENTINEL get-master-addr-by-name mymaster

1) "127.0.0.1"

2) "6380"

# Zaključak

U radu je opisan jedan od problema koji se tiče svih baza podataka, a i servera uopšte, a to je problem visoke dostupnosti prilikom otkaza. Najbolje rešenje je korišćenje više servera kako bi se moglo preuzeti u slučaju otkaza jednog od njih. Predstavljeno je rešenje na primeru Redis baze podataka, gde se za automatsko pokretanje procesa oporavka koristi Redis sentinel. Dat je primer konfigurisanja redis instanci i redis sentinela koje nadgledaju napravljene mastere i replike. Na kraju je dat i primer oporavka

# Reference

1. <https://medium.com/@amila922/redis-sentinel-high-availability-everything-you-need-to-know-from-dev-to-prod-complete-guide-deb198e70ea6>
2. <https://www.tecmint.com/setup-redis-high-availability-with-sentinel-in-centos-8/>
3. <https://redis.io/topics/sentinel>
4. <https://medium.com/@petehouston/install-and-config-redis-on-mac-os-x-via-homebrew-eb8df9a4f298>