Отчет по балансировке.

1. В задании через docker-compose настроен кластер следующим образом:  
- **3 экземпляра postgresql, с настроенной репликацией.**  
- **HAProxy в качестве балансировшика перед БД:**  
 frontend pg\_frontend

bind \*:5432

mode tcp

default\_backend pg\_backend

backend pg\_backend

mode tcp

balance roundrobin

server pg1 pg-1:5432 check

server pg2 pg-2:5432 check

server pg3 pg-3:5432 check  
- **3 экземпляра приложения social\_net, у которых в качестве адреса БД указан адрес Haproxy**

- **nginx, расположенный перед приложениями. Является входной точкой для них, запросы раскидывает round robin`ом. Слушает 80 порт локально:**  
http {

upstream backend {

server node1:8083 max\_fails=1 fail\_timeout=10s;

server node2:8083 max\_fails=1 fail\_timeout=10s;

server node3:8083 max\_fails=1 fail\_timeout=10s;

keepalive 64;

}

server {

listen 80;

location / {

proxy\_pass http://backend;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto $scheme;

}

}

}

2. Исходные данные теста:  
- В БД залиты 30тыс. Строк. Индекса нет.  
social\_net=# select count(\*) from social\_net.users;

count

-------

30302

(1 строка)

- Нагрузку запускаем с помощью утилиты Hey:  
$ hey -c 5 -n 10000 -m GET '[http://localhost:80/user/search?first\_name=%D0%98%D0%B2%D0%B0%D0%BD&last\_name=%D0%90%D0%BD%D1%82%D0%BE%D0%BD%D0%BE](http://localhost:80/user/search?first_name=Иван&last_name=Антоно)'

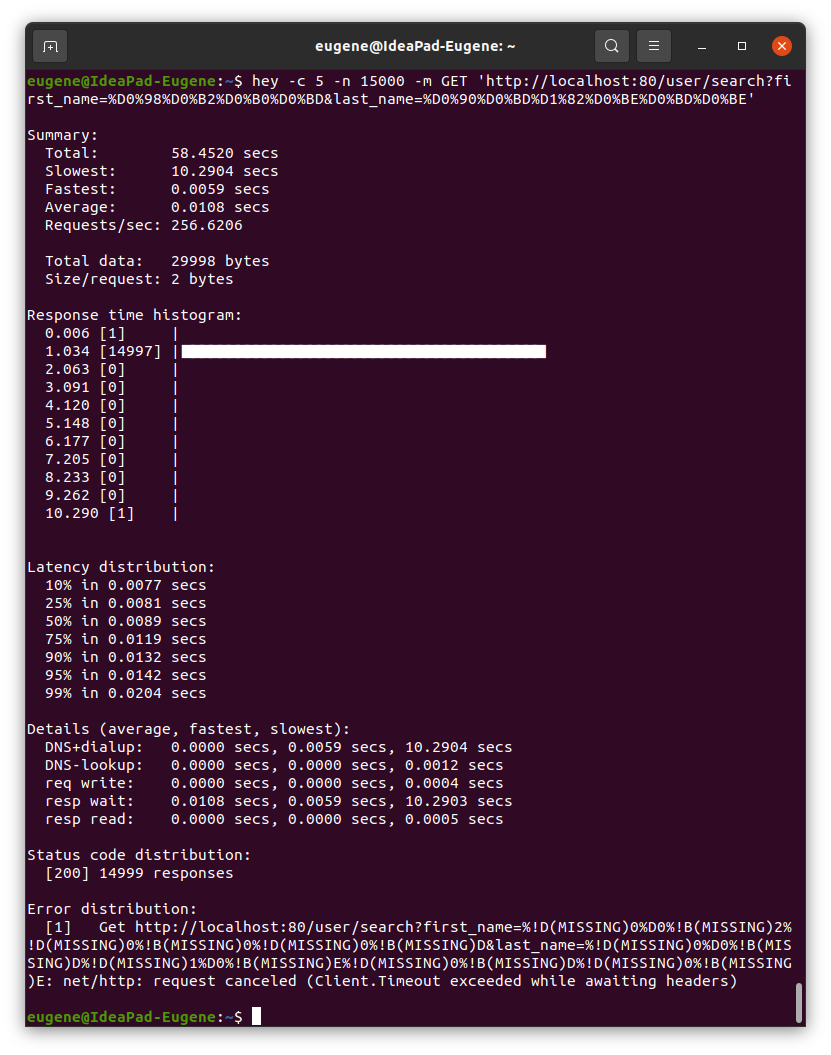
- Выполняем последовательно с паузой 2 команды:

$ docker stop social\_net\_3

$ docker stop postgres\_2

3. Результаты теста:  
- Haproxy успешно отловил падение инстанса postgres, а nginx — приложения.

- За время теста 1 запрос отвалился по таймауту.

Рис. 1. Результат нагрузки.

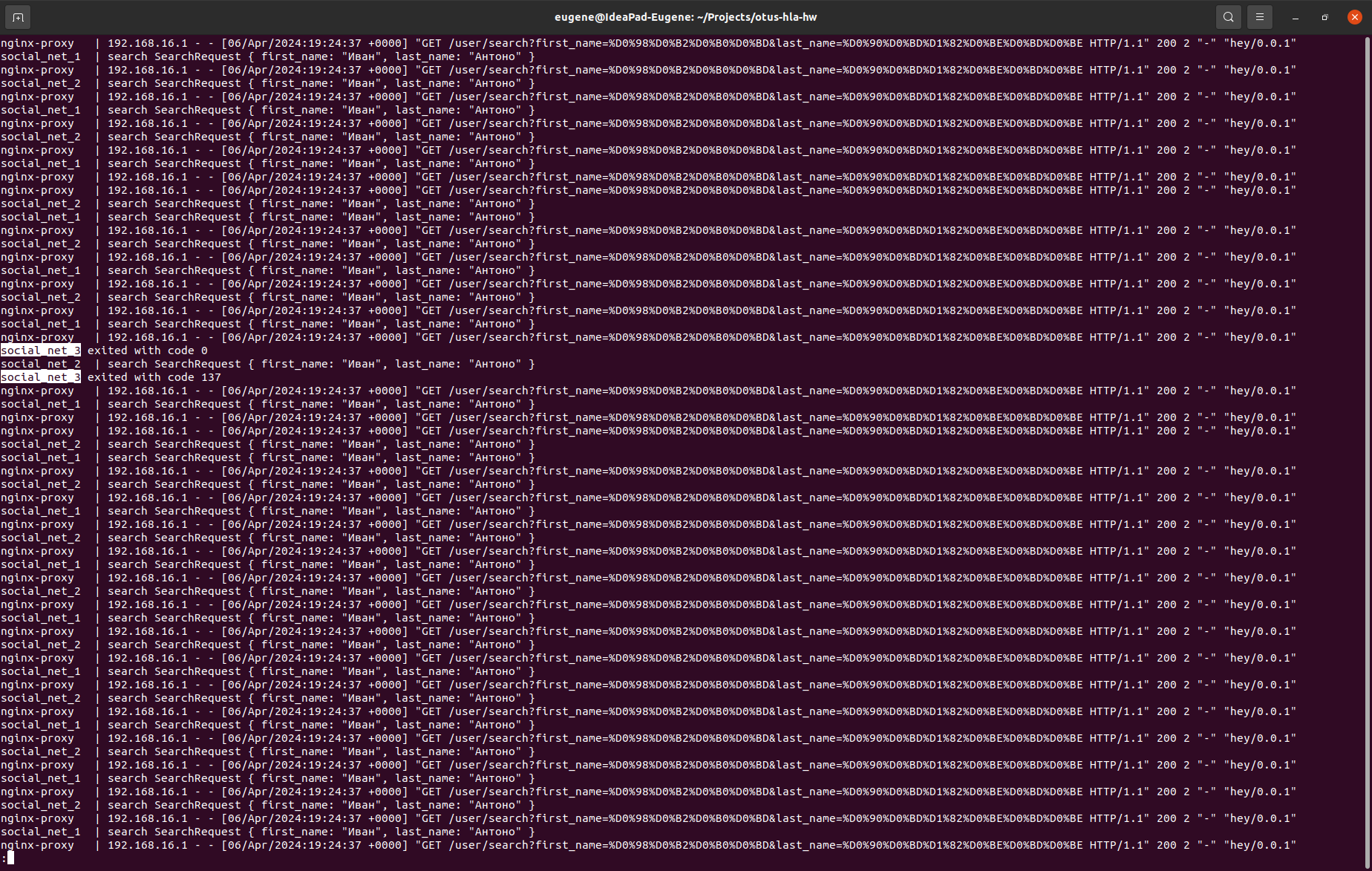


Рис. 2. Момент падения 1 инстанса сервиса.

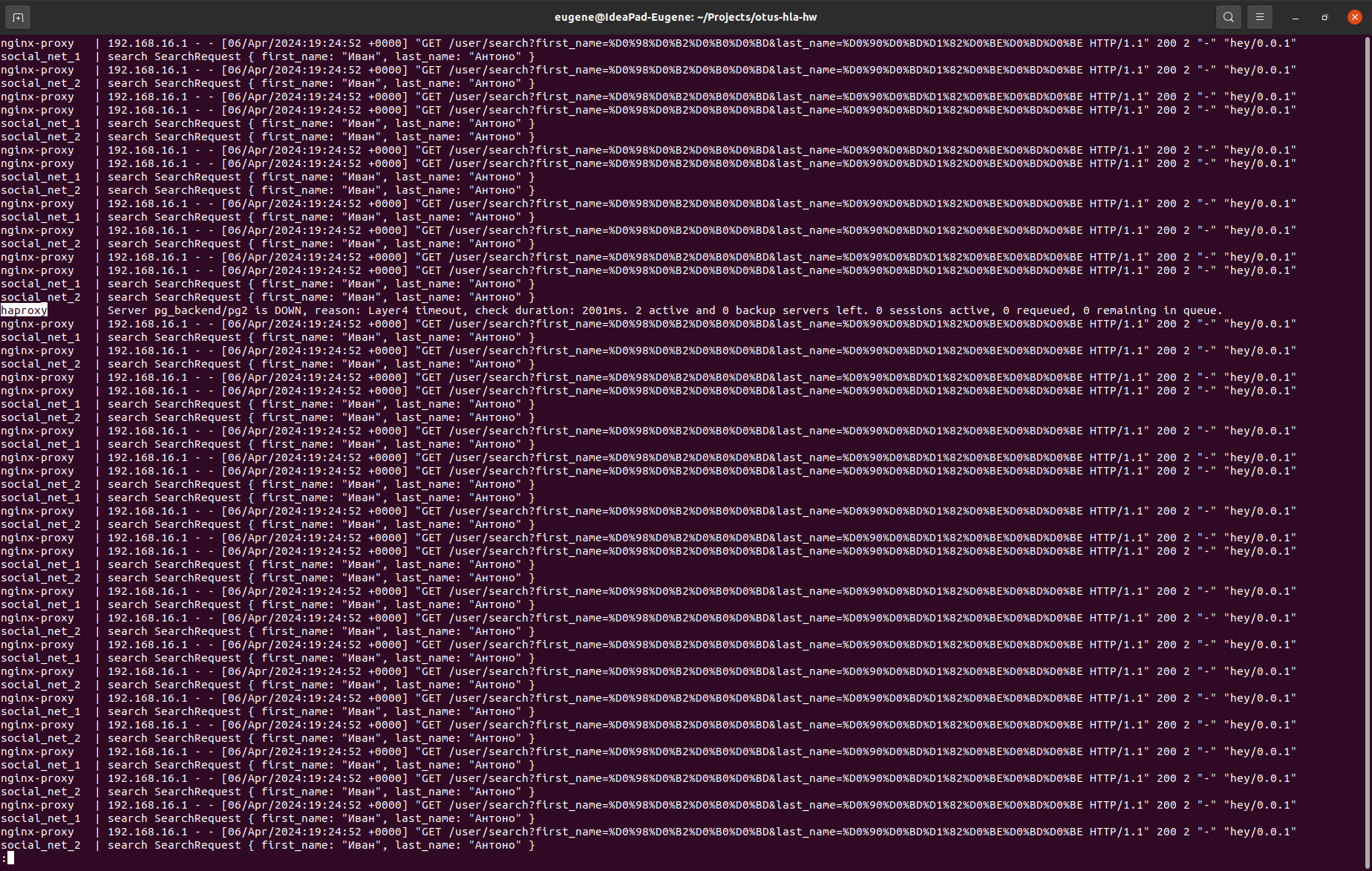


Рис. 3. Момент падения инстанса БД. Заметил только haproxy