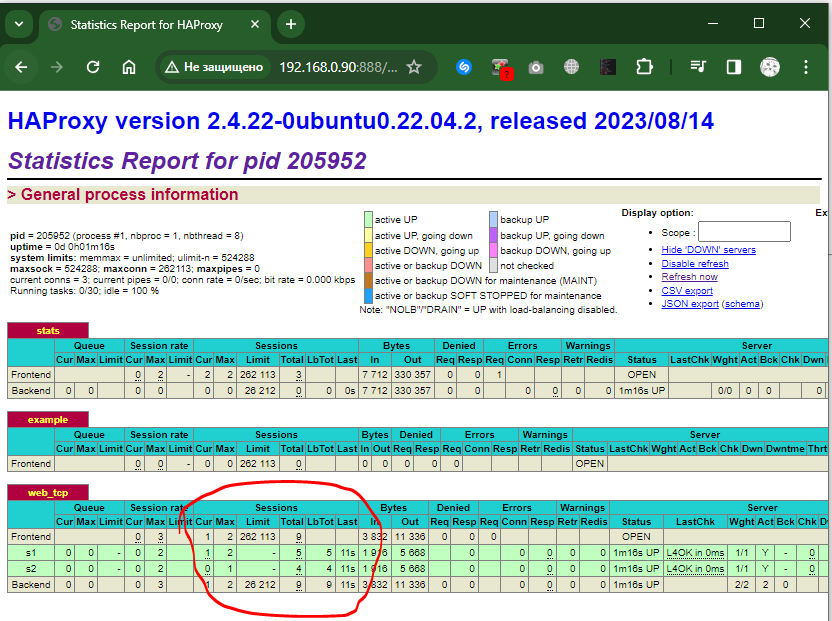
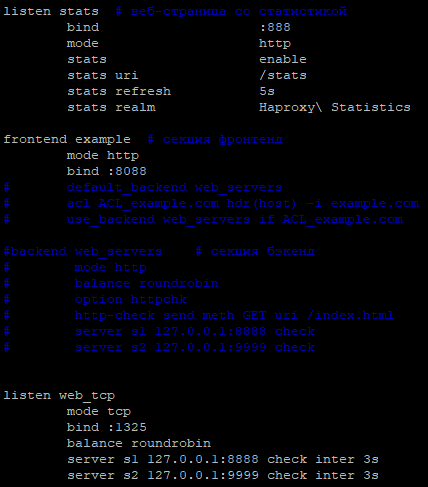
# **Задание 1**

* Запустите два simple python сервера на своей виртуальной машине на разных портах
* Установите и настройте HAProxy, воспользуйтесь материалами к лекции по [ссылке](https://github.com/netology-code/sflt-homeworks/blob/main/2)
* Настройте балансировку Round-robin на 4 уровне.
* На проверку направьте конфигурационный файл haproxy, скриншоты, где видно перенаправление запросов на разные серверы при обращении к HAProxy.

# **Решение 1**



По сессиям видно что roundrobin работает.



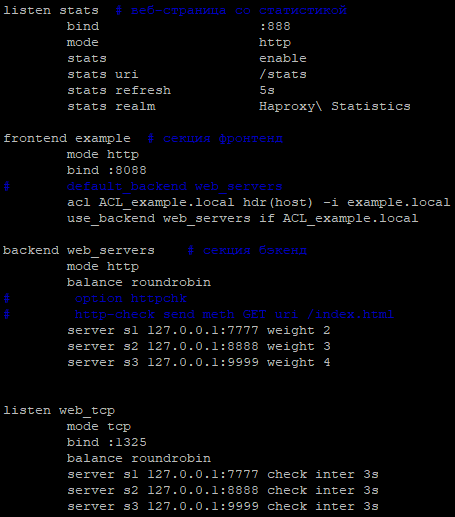
За ненадобностью (по заданию) закоментил L7 часть.

# **Задание 2**

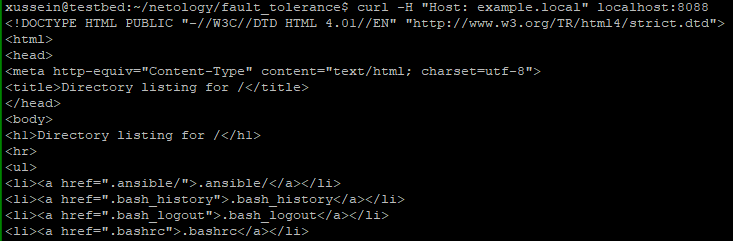
* Запустите три simple python сервера на своей виртуальной машине на разных портах
* Настройте балансировку Weighted Round Robin на 7 уровне, чтобы первый сервер имел вес 2, второй - 3, а третий - 4
* HAproxy должен балансировать только тот http-трафик, который адресован домену example.local
* На проверку направьте конфигурационный файл haproxy, скриншоты, где видно перенаправление запросов на разные серверы при обращении к HAProxy c использованием домена example.local и без него.

# **Решение 2**

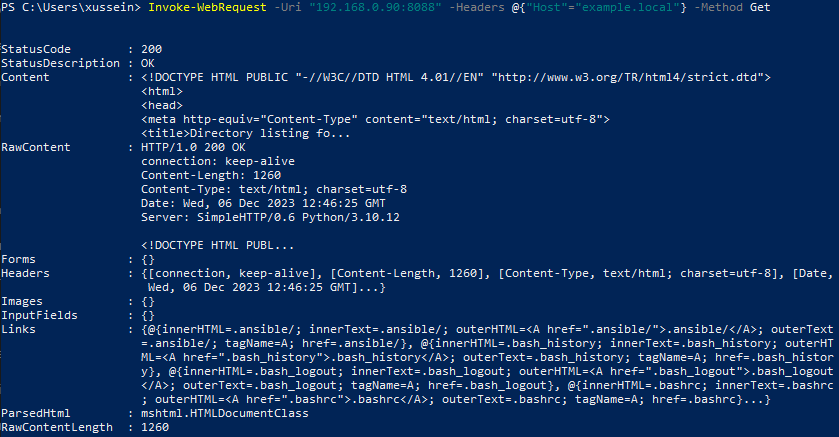
Значимый кусок конфига:



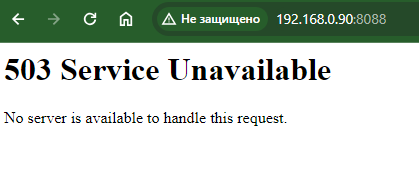
Закинем запрос на сервер через локалхост:



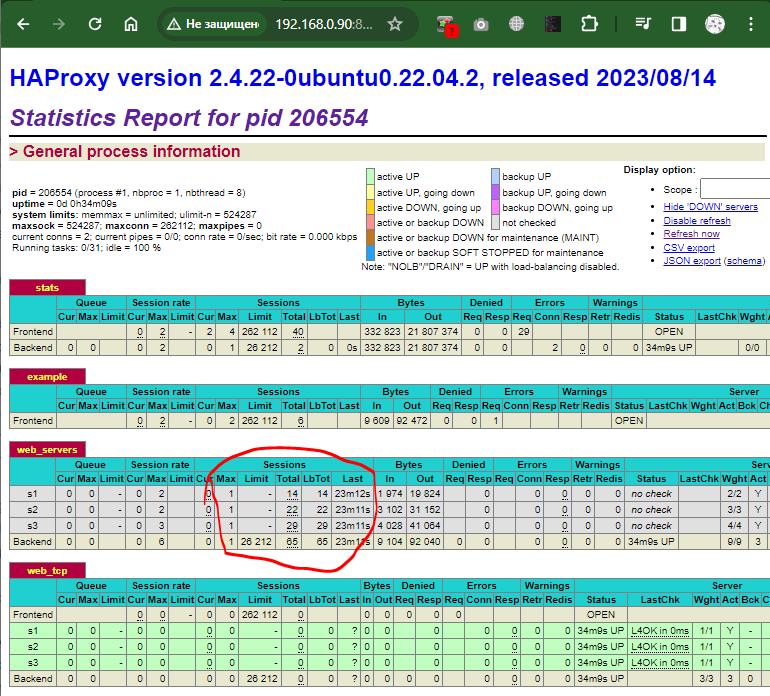
Закинем запрос на сервер через сеть:



Если пробуем просто по адресу постучаться, без правильного заголовка:



И в stats видно следующее:



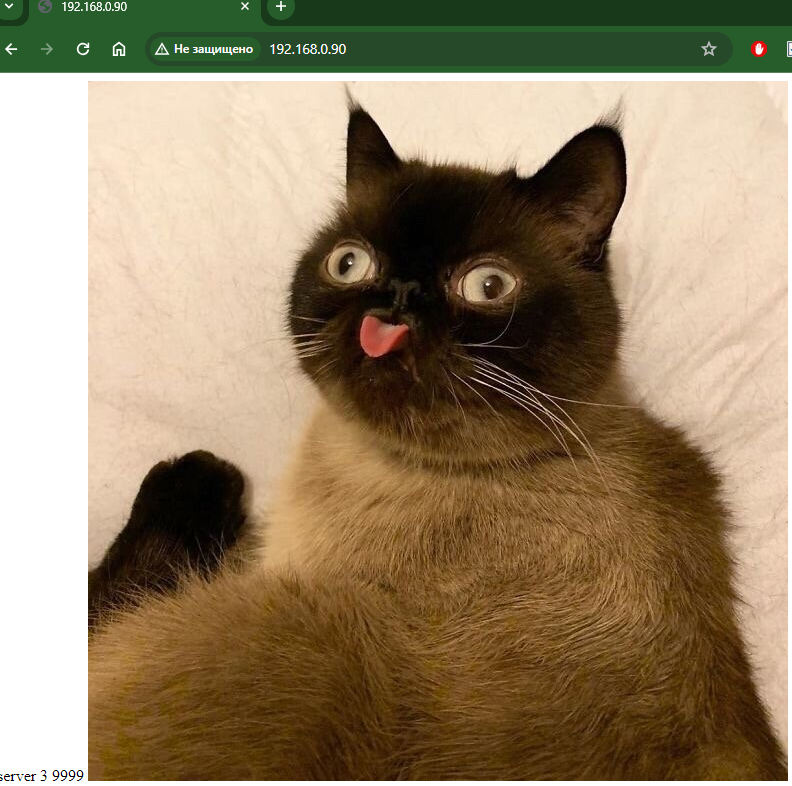
Видно, что количество сессий разное.

# **Задание 3**

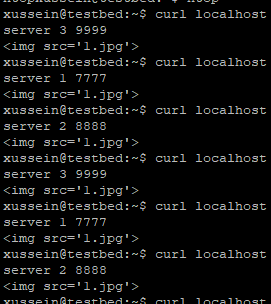
* Настройте связку HAProxy + Nginx как было показано на лекции.
* Настройте Nginx так, чтобы файлы .jpg выдавались самим Nginx (предварительно разместите несколько тестовых картинок в директории /var/www/), а остальные запросы переадресовывались на HAProxy, который в свою очередь переадресовывал их на два Simple Python server.
* На проверку направьте конфигурационные файлы nginx, HAProxy, скриншоты с запросами jpg картинок и других файлов на Simple Python Server, демонстрирующие корректную настройку.

# **Решение 3**

Для каждого сервера был создан index.html, содержащим имя сервера, порт и картинку:



Сервер отрабатывает как положено (roundrobin):



conf файл для сайта:

1. server {
2. listen 80;
3. server\_name netology.example.local;
4. access\_log /var/log/nginx/netology.example.local.log;
5. error\_log /var/log/nginx/netology.example.local;
6. location / {
7. proxy\_pass http://localhost:1325;
8. }
9. location ~ \.(jpg)$ {
10. root /var/www/html;
11. }
12. }

conf файл HAProxy:

1. listen stats # веб-страница со статистикой
2. bind :888
3. mode http
4. stats enable
5. stats uri /stats
6. stats refresh 5s
7. stats realm Haproxy\ Statistics
8. frontend example # секция фронтенд
9. mode http
10. bind :8088
11. # default\_backend web\_servers
12. # acl ACL\_example.local hdr(host) -i example.local
13. # use\_backend web\_servers if ACL\_example.local
14. #backend web\_servers # секция бэкенд
15. # mode http
16. # balance roundrobin
17. # option httpchk
18. # http-check send meth GET uri /index.html
19. # server s1 127.0.0.1:7777 weight 2 check
20. # server s2 127.0.0.1:8888 weight 3 check
21. # server s3 127.0.0.1:9999 weight 4 check
22. listen web\_tcp
23. mode tcp
24. bind :1325
25. balance roundrobin
26. server s1 127.0.0.1:7777 check inter 3s
27. server s2 127.0.0.1:8888 check inter 3s
28. server s3 127.0.0.1:9999 check inter 3s

# **Задание 4**

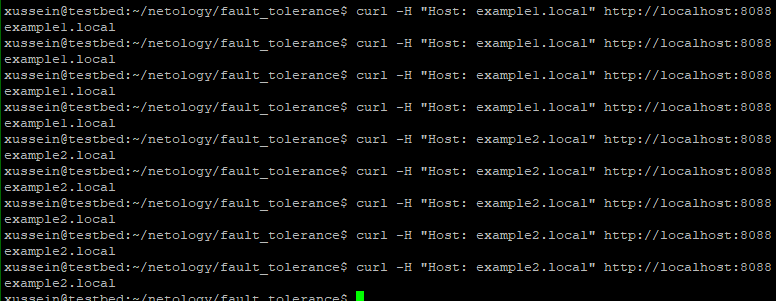
* Запустите 4 simple python сервера на разных портах.
* Первые два сервера будут выдавать страницу index.html вашего сайта example1.local (в файле index.html напишите example1.local)
* Вторые два сервера будут выдавать страницу index.html вашего сайта example2.local (в файле index.html напишите example2.local)
* Настройте два бэкенда HAProxy
* Настройте фронтенд HAProxy так, чтобы в зависимости от запрашиваемого сайта example1.local или example2.local запросы перенаправлялись на разные бэкенды HAProxy
* На проверку направьте конфигурационный файл HAProxy, скриншоты, демонстрирующие запросы к разным фронтендам и ответам от разных бэкендов.

# **Решение 4**

Конфиг HAProxy:

1. listen stats # веб-страница со статистикой
2. bind :888
3. mode http
4. stats enable
5. stats uri /stats
6. stats refresh 5s
7. stats realm Haproxy\ Statistics
8. frontend example # секция фронтенд
9. mode http
10. bind :8088
11. # default\_backend web\_servers
12. acl ACL\_example1.local hdr(host) -i example1.local
13. use\_backend web\_servers\_1 if ACL\_example1.local
14. acl ACL\_example2.local hdr(host) -i example2.local
15. use\_backend web\_servers\_2 if ACL\_example2.local
16. backend web\_servers\_1 # секция бэкенд
17. mode http
18. balance roundrobin
19. option httpchk
20. http-check send meth GET uri /index.html
21. server s1 127.0.0.1:7777 check
22. server s2 127.0.0.1:8888 check
23. backend web\_servers\_2 # секция бэкенд
24. mode http
25. balance roundrobin
26. option httpchk
27. http-check send meth GET uri /index.html
28. server s1 127.0.0.1:6666 check
29. server s2 127.0.0.1:9999 check

Проверяем через curl хосты:



Распределение нагрузки при этом выглядит так:

