

Task 1

1.

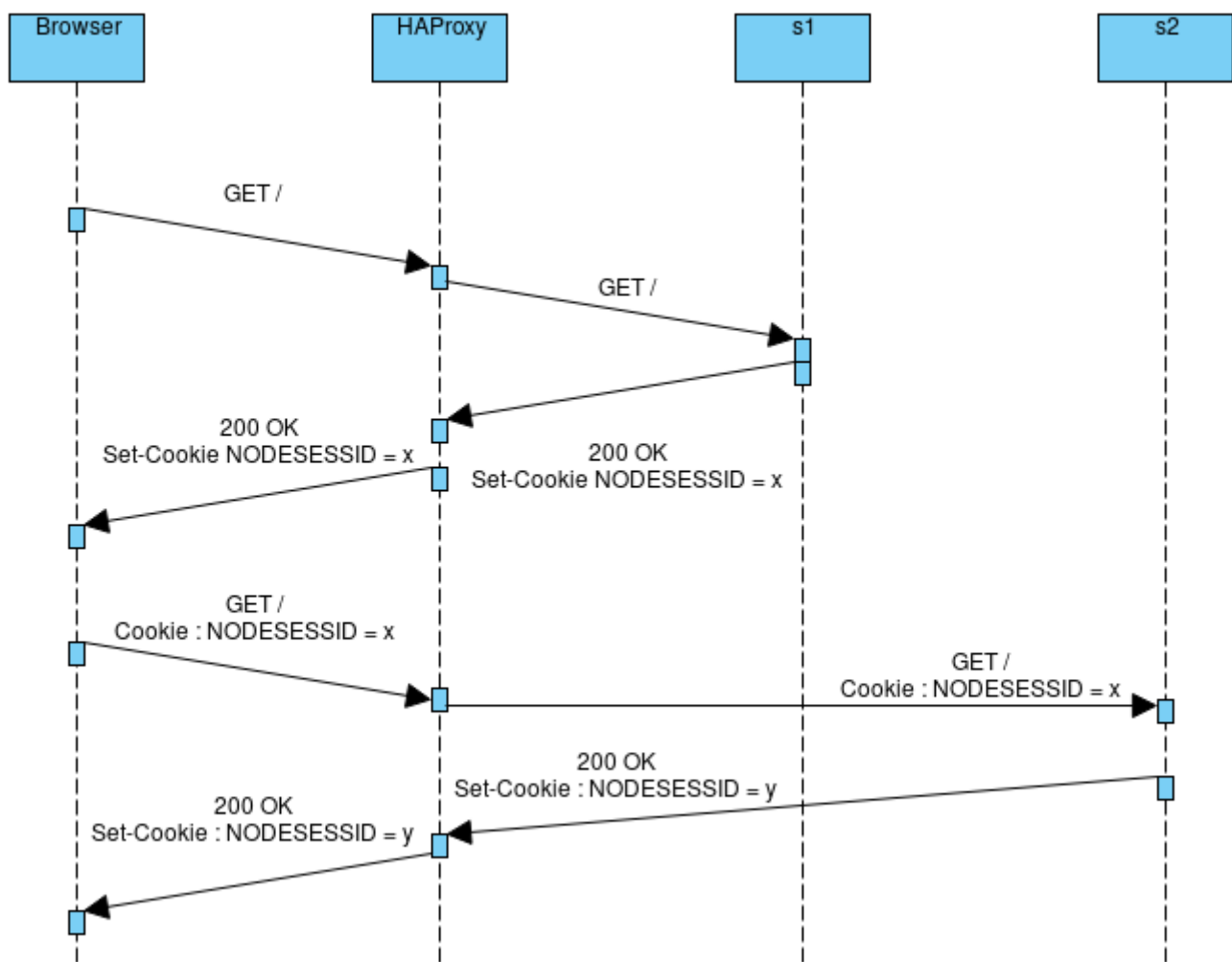
We can see that the proxy handles requests like this : It will address the requests on each server sequentially. It means that one time it will be sent to one server and the next request coming will go threw the other server. The NODESESSID is changing at each request which is not so surprising because each time we visit the site or actualize we have an ID which is invalid for the server requested (because the precedent response came from the other server).

2.

We expect the load balancer to address requests coming in with a session ID to the server who knows this session (so the server who created the session). If not, the session management can't be done.

3.

Here is our sequence diagram for task 1 (roundrobin). This sequence diagram shows 2 requests by the same user, we see that the behaviour is not correct because both requests don't go on the same server as expected.



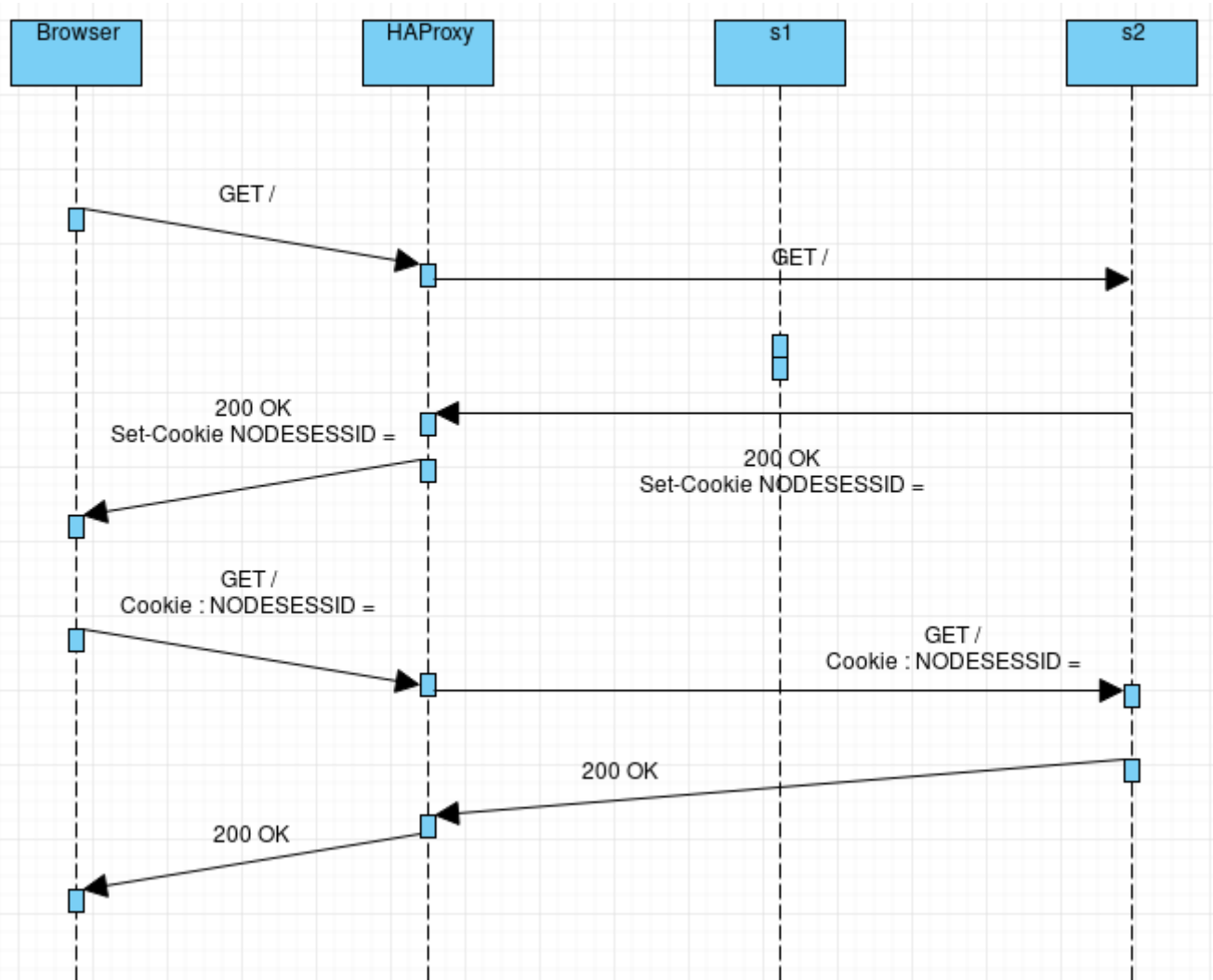
4.

Here is the test in JMeter, we can see that assertions are false because the counters doesn't increment, in fact th server don't know the users coming because of balancing strategy used here.




5.

Here we see the sequence diagram for this task, we can see that because there's only one server running it responds and knows each user (because he serves all the requests), and so the counter increments correctly.



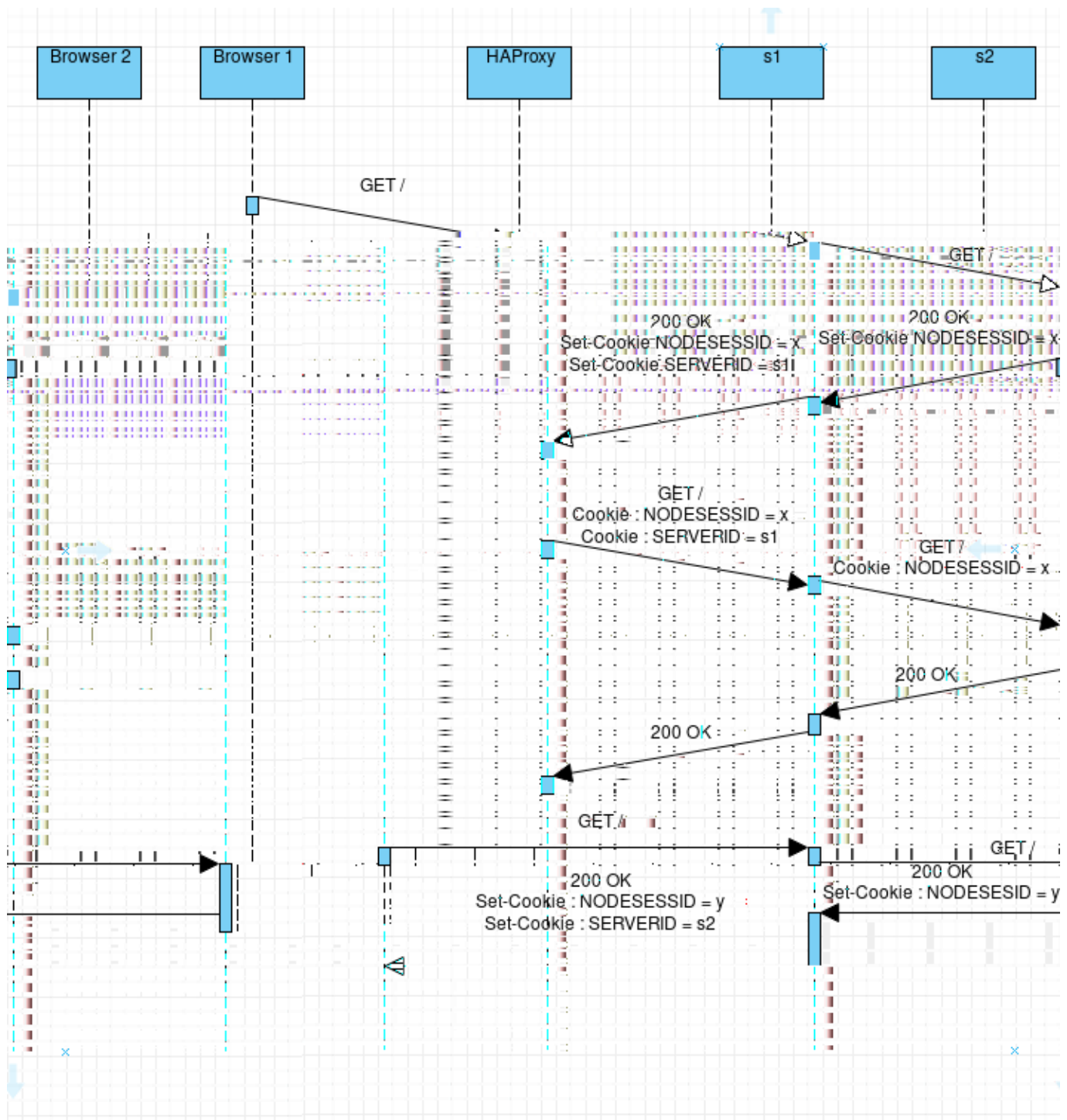
We can see the JMeter report too :

And if we refresh this page we keep the same server again because the SERVERID cookie is well set.

 sticky session 4

4.

We can see here the diagram of the situation where 1 browser refresh the page and what happens if another browser opens a connection :



We can see that the SERVERID cookie is set at the HAProxy level, and it's removed at this level too. And because of the roundrobin if another browser connects to the infra it will be directed to s2 if s1 was the last to respond. This cookie is used by the HAProxy to redirect correctly to the right server.

5.

Here is the summary report of this task :

Summary Report

Name: Summary Report

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only: ☐ Errors ☒ Successes

Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	1000	9	1	60	16.38	0.00%	81.7/sec	29.70	18.67	372.1
S1 reached	1000	0	0	1	0.38	0.00%	81.8/sec	0.00	0.00	.0
TOTAL	2000	5	0	60	12.56	0.00%	163.5/sec	29.70	18.67	186.0

We can explain the behaviour because how JMeter handles the cookies, because there's only one thread group and so one user. So that's normal that we just have one server who responds (because of the SERVERID Cookie).

7.

We can see the result of the manipulations we have done, in fact we added one "user". So the second user when doing his request he's redirected to the other server and all of these future requests will too (because of the cookie).

Task 3

1.

HAProxy

Statistics Report for pid 10

> General process information

pid = 10 (process #1, nproc = 1)
uptime = 5d 0h34m07s
systemd Units: nproc@.nproc = unlimited; utilmap = 4064
maxrss = 4064; maxconn = 2000; maxpipes = 0
current ports = 1; current pipes = 0; conn rate = 0/sec
Running tasks: 1/0 (0% = 0%)

active UP
active UP going down
active DOWN going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
Note: "NOLEBYDRAIN" = UP with load balancing disabled

Display option:
• Scope
• Hide DOWN servers
• Refresh now
• CSV export

External resources:
• Edit my site
• Uploads / CLD
• Online manual

General		Session rate		Sessions		Bytes		Errors		Warnings		Status		Server													
Queue	Cur	Max	Limit	Cur	Max	Limit	Total	Limit	Last	In	Out	Req	Resp	Req	Conn	Resp	Rate	Realis	Status	LastChk	Weight	Act	Back	Chk	Down	Downtime	Thrtle
Frontend	0	1	-	1	2	2 000	16			272 143	7 577 387	0	0	0	0	0	0	0	OPEN			0	0	0	0	0	

Backend		Session rate		Sessions		Bytes		Errors		Warnings		Status		Server													
Queue	Cur	Max	Limit	Cur	Max	Limit	Total	Limit	Last	In	Out	Req	Resp	Req	Conn	Resp	Rate	Realis	Status	LastChk	Weight	Act	Back	Chk	Down	Downtime	Thrtle
Frontend	0	1	-	0	3	2 000	5			62 161	40 152	0	0	0	0	0	0	0	OPEN								

Nodes		Session rate		Sessions		Bytes		Errors		Warnings		Status		Server													
Queue	Cur	Max	Limit	Cur	Max	Limit	Total	Limit	Last	In	Out	Req	Resp	Req	Conn	Resp	Rate	Realis	Status	LastChk	Weight	Act	Back	Chk	Down	Downtime	Thrtle
s1	0	0	-	0	5	0	1	-	21	2	12m27s	16 057	11 305	0	0	0	0	0	33m57s UP	L70x200 in 4ms	1	Y	-	1	1	4s	-
s2	0	0	-	0	5	0	1	-	71	3	11m27s	47 104	28 846	0	0	0	0	0	34m1s UP	L70x200 in 2ms	1	Y	-	0	0	0s	-
Backend	0	0	-	0	0	0	1	200	99	5	11m27s	62 161	40 152	0	0	0	0	0	34m1s UP		2	2	0	0	0	0s	

JSON

Données brutes

En-têtes

Enregistrer Copier Tout réduire Tout développer Filtre le JSON

hello: "world!"

ip: "192.168.42.22"

host: "66c5c8308b0e"

tag: "s2"

sessionViews: 6

id: "UxPOe0g9--4GcZt3RdiCQnE6JmvmkatYa"

We can see in the screenshot above that it is the node "s2" that answers.

2.

5 / 9

HAProxy

Statistics Report for pid 10

> General process information

pid = 10 (process #1, nbproc = 1)
 uptime = 501.1h0m17s
 system load: maxconn = unlimited, ulimit = 4044
 maxconn = 4044, maxconn = 2000, maxconn = 0
 current conn = 5, current pipes = 500, open rate = 0/sec
 Running tasks: 1/10, rate = 99 %

active UP
 active UP, going down
 active DOWN, going up
 active or backup DOWN
 active or backup DOWN for maintenance (MAINT)
 active or backup SOFT STOPPED for maintenance
 Note: "NOLEAF DRAIN" = UP with load-balancing disabled

Display option:

Scope
 Hide DOWN status
 Hide DOWN state
 Hide error

External resources:

HAProxy
 HAProxy 1.8.18
 HAProxy manual

state		Queue		Session rate		Sessions		Bytes		Denied		Errors		Warnings		Status		LastChk		Server		Down		Downtime		Throttle	
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Act	Back	Chk	Down	Downtime	Throttle	
Frontend	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											
Backend																											

We pass the node "s2" on DRAIN mode, it become blue "soft stopped" on the state page.

3.

JSON	Données brutes	En-têtes
Enregistrer	Copier	Tout réduire
Tout développer	Filter le JSON	
hello:	"world!"	
ip:	"192.168.42.22"	
host:	"66c5c8308b0e"	
tag:	"s2"	
sessionViews:	22	
id:	"UxPOe0g9--4GcZt3RdicQnE6JmVkatYa"	

It's the same node than answer our request. In DRAIN mode, all the new traffic will be redirected to the other nodes, but the current session continue to make request to the node in DRAIN mode.

4.

```
{"hello":"world!","ip":"192.168.42.11","host":"d2e8afe714f3","tag":"s1","sessionViews":8,"id":"q9KM156T2unQk11K45rSdKZW186a07Y"}
```

In the new browser , we start a new session and we reach the node "s1".

5.

```
{"hello":"world!","ip":"192.168.42.11","host":"d2e8afe714f3","tag":"s1","sessionViews":1,"id":"L1rfRh7i2LJ4bSqXVbOxQaIrgeuOvFFs"}
```

We clear the cookies on the new browser and we still reach the node "s1". Because "s2" is DRAIN mode, the new session can't reach the "s2". The new browser can only reach "s1".

JSON	Données brutes	En-têtes
Enregistrer	Copier	Tout réduire
Tout développer	Filter le JSON	
hello:	"world!"	
ip:	"192.168.42.22"	
host:	"66c5c8308b0e"	
tag:	"s2"	
sessionViews:	29	
id:	"UxPOe0g9--4GcZt3RdicQnE6JmVkatYa"	

6.

Statistics Report for pid 10

> General process information

[illegible]

It balanced sequentially between the two nodes.

```
{ "hello": "world!", "ip": "192.168.42.22", "host": "66c5c8308b0e", "tag": "s2", "sessionViews": 4, "id": "UPtIM ZOo3iSkHT7lso96WwMAn0AvYM0" }
```

The screenshot shows the JSON Editor interface. At the top, there are tabs for 'JSON', 'Données brutes', and 'En-têtes'. Below these are action buttons: 'Enregistrer', 'Copier', 'Tout réduire', 'Tout développer', and a search icon with the text 'Filtrer le JSON'. The main area displays a JSON object:

```
{  "hello": "world!",  "ip": "192.168.42.22",  "host": "66c5c8308b0e",  "tag": "s2",  "sessionViews": 33,  "id": "UxPOe0g9--4GczT3RdicQNE6JmvykatYa"}
```

7.

Statistics Report for pid 10

> General process information

pid = 10 (process #1, nprocs = 1)
 uptime = 5d 1h16m1s
 system limits: memmax = unlimited, ulimits = 4044
 maxprocs = 4044, maxconn = 2000, maxpipes = 0
 current pipes = 1, current pipes = 5/0, open files = 1366
 Running tasks: 1/9, idle = 100 %

active UP going down
 active UP going up
 active DOWN going up
 active or backup DOWN
 active or backup DOWN for maintenance (MAINT)
 active or backup SOFT STOPPED for maintenance

backup UP going down
 backup UP going down
 backup DOWN going up
 not checked
 active or backup DOWN for maintenance (MAINT)
 active or backup SOFT STOPPED for maintenance

Note: "NOLE" or "DRAIN" = UP with load-balancing disabled.

Display option:

- Scope
- Host GROUP names
- Refresh rate
- Cell format

External resources:

- External API
- External API
- External API

On MAINT mode, all the traffic is redirected to the other nodes, including existing sessions. The node "s2" can't be reached by request.

```
{"hello":"world!","ip":"192.168.42.11","host":"d2e8afe714f3","tag":"s1","sessionViews":1,"id":"yEsiGQ_Uz6Wgf-waB6wiGd7T6u_FmHlWw"}
```

The new browser reach the node "s1". If we clear the cookies, we still only reach "s1".

JSON	Données brutes	En-têtes
Enregistrer	Copier	Tout réduire
	Tout développer	Filtrer le JSON
hello:	"world!"	
ip:	"192.168.42.11"	
host:	"d2e8afe714f3"	
tag:	"s1"	
sessionViews:	1	
id:	"RWmemHV5130leKU_EvrES4-QRsIeerps"	

The first browser is not anymore on the node "s2". And has a new session on the node "s1".

Task 4

1.

We have been asked to do a run for base data with the base config, so we set the delays to 0 for both servers (we took the JMeter conf. used for Task 2, 2 thread groups):

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	2000	16	2	58	19.36	0.00%	102.0/sec	37.08	23.41	372.1
S2 reached	1000	0	0	1	0.37	0.00%	52.0/sec	0.00	0.00	.0
S1 reached	1000	0	0	2	0.36	0.00%	52.4/sec	0.00	0.00	.0
TOTAL	4000	8	0	58	15.90	0.00%	204.1/sec	37.08	23.41	186.0

2.

We set a 250ms delay to s1 :

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	2000	763	1	2038	828.15	0.00%	1.3/sec	0.48	0.30	372.1
S2 reached	1000	0	0	4	0.51	0.00%	47.6/sec	0.00	0.00	.0
S1 reached	1000	0	0	6	0.50	0.00%	39.6/min	0.00	0.00	.0
TOTAL	2000	763	1	2038	828.15	0.00%	1.3/sec	0.48	0.30	372.1

We can see with this run taht it took a really long time (25min) because $1000 \times 0.25s$, but it works well in fact. What happens it's that the first user takes the connection to s1 and it will be long because the cookie specify s1 for each request. And the second user go to s2 without problems and takes each time s2 because of the cookie.

3.

Proof that we set correctly the delay :

POST /delay	2.276 ms - 47									
HEAD /	1.408 ms - 129									
HEAD /	1.500 ms - 129									
HEAD /	2506.530 ms - 129									
HEAD /	1.318 ms - 129									
HEAD /	1.925 ms - 129									
HEAD /	2508.473 ms - 129									
HEAD /	1.621 ms - 129									
HEAD /	1.685 ms - 129									
[WARNING] 335/123344 (11) : Server nodes/s1 is DOWN, reason: Layer7_timeout, check duration: 2001ms. 1 active and 0 backup servers left. 0 sessions active, 0 requested, 0 remaini										
HEAD /	2508.326 ms - 129									
HEAD /	1.722 ms - 129									
HEAD /	1.542 ms - 129									
HEAD /	2508.321 ms - 129									
HEAD /	1.358 ms - 129									

We can see on the proof that HA detects s1 as DOWN. We set a 2500ms delay to s1 :

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	2000	15	2	58	19.09	0.00%	103.9/sec	37.76	23.94	372.1
S2 reached	2000	0	0	2	0.38	0.00%	104.0/sec	0.00	0.00	.0
TOTAL	4000	8	0	58	15.63	0.00%	207.8/sec	37.75	23.93	186.0

We see that apparently the load balancer managed to see that the s1 server was too slow and directed all requests to s2 (see proof above). But that's really weird because this timeout for connecting to a server is defined as 5000ms.

4.

No we didn't have any error on these tasks, we think that the load balancer is smart enough to balance quite well, and we didn't make enough requests to cause an error.

5.

After doing the weight config we have this behaviour in the JMeter tests :

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	2000	155	1	570	149.98	0.00%	6.7/sec	2.42	1.52	372.1
S1 reached	1000	0	0	1	0.36	0.00%	3.3/sec	0.00	0.00	.0
S2 reached	1000	0	0	1	0.36	0.00%	57.8/sec	0.00	0.00	.0
TOTAL	4000	77	0	570	131.54	0.00%	13.3/sec	2.42	1.52	186.0

Nothing change from the 2. apart of the time taken. That's because the weights doesn't change anything for 2 users.

6.

Now the weights takes effects because it's like we have another user each time, because we clean cookies for each iteration. So the weights influence the behaviour of the load balancer in this way :

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
GET /	2000	373	3	1032	291.74	0.00%	5.2/sec	2.81	0.60	552.6
S1 reached	1334	0	0	1	0.39	0.00%	3.5/sec	0.00	0.00	.0
S2 reached	666	0	0	1	0.37	0.00%	1.7/sec	0.00	0.00	.0
TOTAL	4000	186	0	1032	278.17	0.00%	10.4/sec	2.81	0.60	276.3